ISSN 2464–4617 (print) ISSN 2464–4625 (CD-ROM)

Computer Science Research Notes

CSRN 3001

(Eds.)

- **Vaclav Skala**
  University of West Bohemia, Czech Republic

28. International Conference in Central Europe on
Computer Graphics, Visualization and Computer Vision
WSCG 2020
Plzen, Czech Republic
May 18 – 22, 2020

Proceedings

WSCG 2020

Proceedings

ISSN 2464–4617 (print) ISSN 2464–4625 (CD-ROM)

ISSN 2464–4617 (print) ISSN 2464–4625 (CD-ROM)

Computer Science Research Notes

CSRN 3001

(Eds.)

- **Vaclav Skala**
  University of West Bohemia, Czech Republic

28. International Conference in Central Europe on
Computer Graphics, Visualization and Computer Vision
WSCG 2020
Plzen, Czech Republic
May 18 – 22, 2020

Proceedings

WSCG 2020

Proceedings

ISSN 2464–4617 (print) ISSN 2464–4625 (CD-ROM)

Computer Science Research Notes
CSRN 3001

Editor-in-Chief: Vaclav Skala
c/o University of West Bohemia
Univerzitni 8
CZ 306 14 Plzen
Czech Republic
skala@kiv.zcu.cz http://www.VaclavSkala.eu

Managing Editor: Vaclav Skala

Publisher & Author Service Department & Distribution:
Vaclav Skala - UNION Agency
Na Mazinach 9
CZ 322 00 Plzen
Czech Republic
Reg.No. (ICO) 416 82 459

Published in cooperation with the University of West Bohemia
Univerzitní 8, 306 14 Pilsen, Czech Republic

ISSN 2464-4617 (Print) ***ISSN 2464-4625*** (CD/DVD)
ISBN 978-80-86943-35-0 *(*electronic)

WSCG 2020

International Program Committee

Barton,M. (Spain)
Benes,B. (USA)
Benger,W. (Austria)
Birra,F. (Portugal)
Bourke,P. (Australia)
Bujack,R. (USA)
Carvalho,M. (Brazil)
Coquillart,S. (France)
Debelov,V. (Russia)
Drechsler,K. (Germany)
Durikovic,R. (Slovakia)
Feito,F. (Spain)
Feng,J. (China)
Ferguson,S. (U.K.)
Galo,M. (Brazil)
Gdawiec,K. (Poland)
Groeller,E. (Austria)
Gudukbay,U. (Turkey)
Gunther,T. (Switzerland)
Hast,A. (Sweden)
Chaudhuri,D. (India)
Choi,S. (Korea)
Juan,C. (Spain)
Klosowski,J. (USA)
Kurt,M. (Turkey)
Li,X. (USA)
Liu,S. (China)
Lobachev,O. (Germany)
Molla,R. (Spain)
Montrucchio,B. (Italy)
Muller,H. (Germany)
Nasri,A. (UAE)
Oliveira,J. (Portugal)
Pan,R. (China)
Paquette,E. (Canada)
Pedrini,H. (Brazil)
Platis,N. (Greece)
Renaud,C. (France)
Richardson,J. (USA)
Rodrigues,J. (Portugal)
Rojas-Sola,J. (Spain)
Santos,L. (Portugal)
Sarfraz,M. (Kuwait)
Skala,V. (Czech Rep.)
Segura,R. (Spain)
Teschner,M. (Germany)
Tokuta,A. (USA)
Trapp,M. (Germany)
Wu,S. (Brazil)
Wuensche,B. (New Zealand)
Zwettler,G. (Austria)

WSCG 2020

Board of Reviewers

Tokuta,A. (USA)
Toler-Franklin,C. (USA)
Trapp,M. (Germany)
Tytkowski,K. (Poland)
Vanderhaeghe,D. (France)
Wiegreffe,D. (Germany)
Wu,S. (Brazil)
Wuensche,B. (New Zealand)
Yoshizawa,S. (Japan)
Zwettler,G. (Austria)

CSRN 3001

Computer Science Research Notes

WSCG 2020 Proceedings

Contents

Memory-Friendly Deep Mesh Registration

François Le Clerc
InterDigital
975 avenue des Champs Blancs
F-35576 Cesson-Sévigné, France
Francois.LeClerc@InterDigital.com

Hao Sun[1]
Tencent
1081C Hongmei Road
200233 Shanghai, China
mikiyasun@tencent.com

ABSTRACT

Processing 3D meshes using convolutional neural networks requires convolutions to operate on features sampled on non-Euclidean manifolds. To this purpose, spatial-domain approaches applicable to meshes with different topologies locally map feature values in vertex neighborhoods to Euclidean 'patches' that provide consistent inputs to the convolution filters around all mesh vertices. This generalization of the convolution operator significantly increases the memory footprint of convolutional layers and sets a practical limit to network depths on the available GPU hardware. We propose a memory-optimized convolution scheme that mitigates the issue and allows more convolutional layers to be included in a network for a given memory budget. The experimental evaluation of mesh registration accuracy on datasets of human face and body scans shows that deeper networks bring substantial performance improvements and demonstrate the benefits of our scheme. Our results outperform the state of art.

Keywords

Geometric deep learning, convolutional neural networks, shape matching, 3D mesh

1 INTRODUCTION

Mesh registration, aka shape matching, is a key stage of a 3D geometry processing pipeline that provides control on the sampling of vertices and brings all processed meshes to a common representation. Effective registration techniques based on non-rigid Iterative Closest Point approaches have been developed, but their computational cost makes them unsuitable for interactive processing. Besides, landmark annotations are needed to drive the convergence of these algorithms. Recent advances in machine learning and particularly deep learning offer promising prospects for improvements in the field. Inference on a deep network is fast thanks to the availability of high-performance GPU hardware, and learning rather than computing the registration removes the need for landmarking. The challenge for learning-based approaches is to maintain the high levels of accuracy achieved by computational geometry algorithms.

Convolutional neural networks were originally applied to signals regularly sampled on a Euclidean domain, for which the implementation of discrete convolution is straightforward. For signals defined on non-Euclidean mesh surfaces, the convolution operator becomes a position-dependent filter [Bro17a]. When the processed meshes have different topologies, the feature values known at the locations of vertices must be locally mapped to Euclidean 'patches' to provide the inputs to the convolution filters. The patch extraction process incurs additional computations and increased memory requirements. In practice, the network depth is limited by the memory capacity of the available GPU hardware: network architectures proposed in [Mas15a, Mon17a, Ver18a] do not contain more than 3 convolutional layers.

In this paper we quantify the storage space needed for implementing convolutions on feature signals sampled on the surface of a 3D mesh. In addition to the Euclidean case requirements, local geometry information around each vertex must be fed into the network for extracting the patches, and storage must be allocated for intermediate tensors in the computation of convolutions. We propose mitigation schemes for both of these issues to reduce the memory footprint of convolutional layers. For a given memory budget, more of these layers can be squeezed into the network. The shape matching performance of our memory-optimized networks is evaluated on human body and human face mesh datasets with varying resolutions. The results show that the registration accuracy greatly benefits from in-

[1] Work done while the author was with Technicolor R&D, now InterDigital

https://doi.org/10.24132/CSRN.2020.3001.1

creased network depths. Our approach outperforms the state-of-art and achieves close to perfect registration scores on the FAUST dataset.

2 RELATED WORK

We focus our review of past research on approaches most related to our work, and in particular to registration algorithms that can deal with non-isometric deformations. For a comprehensive review of shape matching, we refer the reader to the survey in [Kai04a].

Non-rigid Iterative Closest Point

Early shape matching approaches [Sum04a, Amb07a] were extensions of the Iterative Closest Point method to non-rigid registration, where the deformation between a source and a target mesh is modelled as a set of local affine transforms. The registration is obtained by globally solving for the deformed source vertex positions, with regularization terms to ensure a spatially smooth mapping. Landmark correspondences, typically obtained by manual annotation, are required to initiate the process. [Zel13a] improves the resilience of these methods to large differences in the geometries of the source and target mesh by initiating the computation of the deformation on geometrically simplified meshes. All of these approaches require solving a bulky sparse linear system dimensioned by the number of vertices and faces in the meshes, making them unsuitable for interactive operation except on small meshes.

Functional Maps

Functional maps [Ovs12a] extend the notion of vertex-to-vertex correspondence to mappings between real-valued functions that encode descriptors of the geometry around each vertex. The mapping between functions is expressed as a linear operator on orthogonal bases taken to be the eigenfunctions of the Laplace-Beltrami operator. Solving a large linear system yields the coefficients of this linear operator. [Lit17a] improves on the original approach by leveraging a deep network to jointly compute the functional map correspondence and optimize the functional descriptors.

Learning descriptor-based correspondences

In early machine learning approaches, shape registration is cast as a classification problem and the correspondence is computed from local mesh geometry descriptors. In [Rod14a] correspondence labels for shape vertices are computed from the outputs of a random forest whose split functions are designed based on the Wave Kernel Signatures of the vertices. In [Wei16a] per-pixel descriptors on depth maps of human bodies are learnt as intermediate features of a shape classification network, and the correspondence problem is solved by a closest neighbor search in descriptor space.

Graph Convolutional Neural Networks

Deep learning approaches formulate mesh registration as a classification problem whose outputs for each source mesh vertex are the probabilities of assignment to each of the target mesh vertices. The generalization of Convolutional Neural Networks (CNNs) to signals sampled on non-Euclidean domains (in the most general setting, graphs) can be achieved by defining the convolution in the graph Fourier domain [Bro17a]. However, these spectral-domain approaches, *e.g.* [Kip17a], are inadequate for processing meshes with different topologies as the graph Fourier transform differs for each mesh. Approaches applicable to shape matching operate in the spatial domain by locally mapping feature values in a neighborhood of each vertex on the mesh surface to a Euclidean domain on which a convolution kernel can be defined consistently for all vertices. In [Mas15a, Bos16a, Mon17a] these mappings target local geodesic polar coordinate systems, while in [Ver18a] they are learnt from the convolutional layer inputs.

The adaptation of the convolution operator in spatial-domain graph-CNN approaches incurs additional computations and increased memory requirements that limit in practice the network depth. Based on the intuition that deeper networks could provide better performance, we propose a memory-optimized convolution scheme that allows more convolutional layers to be included in the network for the same memory budget.

3 TECHNICAL APPROACH

3.1 Problem and notations

We address the shape matching problem for input meshes with N vertices by means of a spatial-domain graph-CNN. For simplicity and without loss of generality, we assume minibatches to enclose the feature descriptors of all the vertices of a mesh. We consider a generic convolutional layer of the network with D input channels, E output channels and convolution kernels of size M. The learnable parameters for the layer are the convolution weights $w_m^{d,e}$ with $1 \leq d \leq D$, $1 \leq e \leq E$ and $1 \leq m \leq M$.

3.2 Patch operator

The convolution can be formulated using a "patch operator" [Mas15a, Bro17a] that interpolates a feature signal f sampled at vertex locations on the mesh surface to consistent inputs to the convolution operator, based on data available in local neighborhoods Γ_i of each vertex V_i:

$$D_m(V_i)f = \sum_{j \in \Gamma_i} c_m[\mathbf{u}(V_i,V_j)].f(V_j). \quad (1)$$

In this equation, m indexes the values of the convolution kernel, $\mathbf{u}(V_i,V_j)$ is a local parameterization of the mesh

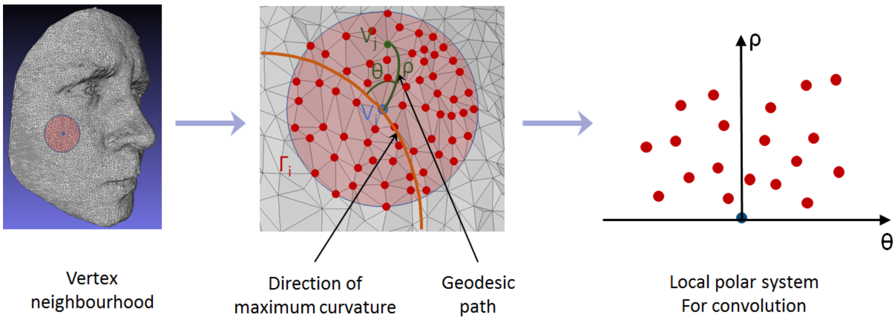


Figure 1: *the patch operator maps vertices in a vertex neighbourhood Γ_i to a Euclidean polar coordinate system*

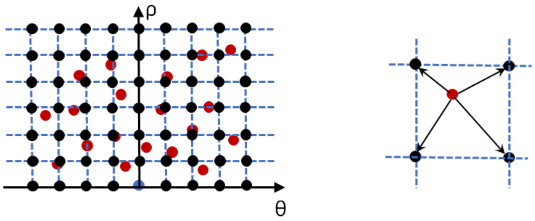


Figure 2: *the convolution kernel support is a uniformly sampled grid in the (ρ,θ) space (black dots). The feature value at each vertex neighbor V_j (red dots) contributes only to the 4 neighboring grid points.*

surface in Γ_i and the c_m weights define the patch operator. The Γ_i can be defined as geodesic, Euclidean or ring neighborhoods. Their size $K_i \overset{\text{def}}{=} |\Gamma_i|$ is a dimensioning parameter for the memory footprint of the convolutional layer. We denote by K the average of K_i over the mesh. Following [Mas15a, Mon17a] we define $\mathbf{u}(V_i,V_j) = (\rho_{ij},\theta_{ij})^T$ as local polar intrinsic coordinate systems where ρ_{ij} is the geodesic distance between V_i and V_j and θ_{ij} the angle between the geodesic path at V_i with a reference direction that we set to the maximum curvature direction at V_i [1] (see Figure 1). For notational simplicity we define $c_{ijm} \overset{\text{def}}{=} c_m[\mathbf{u}(V_i,V_j)]$ and obtain the e^{th} output component of the convolution as

$$g_i^e = \sum_{d=1}^{D} \sum_{j\in\Gamma_i} \sum_{m=1}^{M} c_{i,j,m} w_m^{d,e} f_j^d. \tag{2}$$

In [Mas15a, Mon17a] the patch operator relies on a mixture of Gaussian kernels :

$$c_m(\mathbf{u}) = exp[-\frac{1}{2}(\mathbf{u}-\mathbf{u}_m)^T \begin{pmatrix} \sigma_\rho^2 & 0 \\ 0 & \sigma_\theta^2 \end{pmatrix}^{-1} (\mathbf{u}-\mathbf{u}_m)] \tag{3}$$

with the difference that σ_ρ, σ_θ and $\mathbf{u}_m$ are fixed in [Mas15a] and learnt in [Mon17a]. [Ver18a] remove the hand-crafted parameterization $\mathbf{u}(V_i,V_j)$ of the mesh surface and directly learn the patch operator as a function of the feature inputs to the current layer:

$$c_m(f_i, f_j) \propto exp[\mathbf{a}^T(f_i - f_j) + \mathbf{b}_m]. \tag{4}$$

We opt for the baseline patch operator of [Mas15a] and define the support of the convolution kernel to be a regular grid of $N_\rho \times N_\theta$ points in the local polar geodesic systems around each vertex. To reduce training times we speed up the computation by bi-linearly interpolating the contribution of the feature value at each V_j to its 4 closest points on the grid (Figure 2). Subject to the corresponding restrictions on $|\rho_{ij} - \rho_m|$ and $|\theta_{ij} - \theta_m|$, letting $\Delta\rho$ and $\Delta\theta$ be the spacings between adjacent grid points along each axis, our patch operator is defined as

$$c_m((\rho_{ij},\theta_{ij})^T) = \left(1 - \frac{|\rho_m - \rho_{ij}|}{\Delta\rho}\right)\left(1 - \frac{|\theta_m - \theta_{ij}|}{\Delta\theta}\right) \tag{5}$$

We weight the contributions of vertex features $f(V_j)$ in to the patch operator (1) by a term that approximates the Voronoi area of V_j, thereby giving more importance to feature values that represent larger areas on the mesh surface. Specifically, each triangle a vertex belongs to contributes 1/3 of its area to the vertex weight. The weight value is accumulated over all the triangles. Experimentally we found that this refinement stabilizes the convergence of the training process and slightly improves the registration accuracy.

3.3 Memory optimization

Implementing CNNs on non-Euclidean domains using a patch operator (1) incurs extra complexity: a local parameterization $\mathbf{u}(V_i,V_j)$ of the mesh surface around each V_i is needed, and the convolution involves an extra summation level. We show below that this strongly impacts memory consumption.

To obtain orders of magnitude on the memory footprints of convolutional layers, we consider as a typical use case medium resolution meshes with $N = 15000$ vertices and an average vertex neighborhood size of $K = 300$ that corresponds to the optimum in our experiments on face meshes. We further assume values of $M = 32$,

[1] [Mas15a] does not define a reference direction but computes convolutions for N_θ possible rotations of the convolution filter and retains the maximum value as the output.

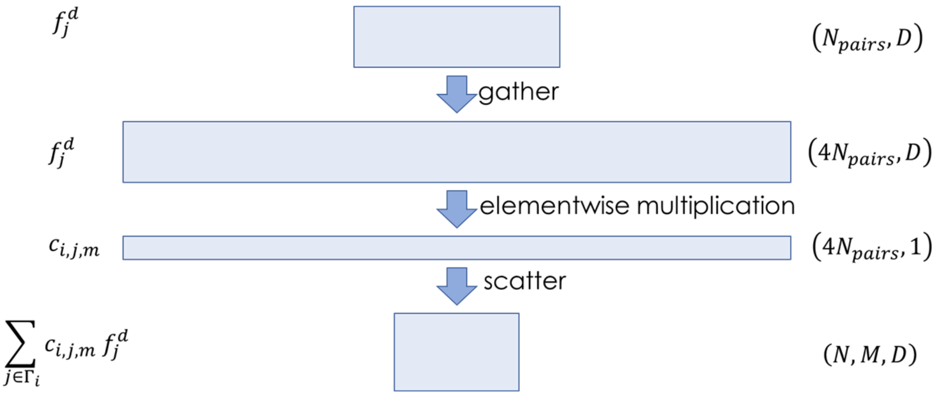


Figure 3: *optimized memory management of variable-sized vertex neighborhoods.*

D = 128, E = 256 that are typical of the experimental setups used in [Mas15a, Mon17a, Ver18a].

The patch operators used in [Mas15a, Mon17a] and in our approach depend on local polar coordinates $\mathbf{u}(V_i, V_j)$ around each V_i on the mesh surface. This information must be fed to the network for extracting the convolution patches around each vertex through the $c_m(\mathbf{u})$ coefficients and is used in each convolutional layer. These coefficients represent in total NKM floating point values for a minibatch of N vertices. In the approach of [Ver18a] the geometrical data needed to compute the patch operator are the indicator functions of neighborhoods Γ_i, NK integer vertex indices whose memory footprint is negligible.

Assuming a 4-byte floating point representation, NKM values amount to 0.58 GB of data in our numerical example. However, this estimation assumes neighborhoods of fixed size K. In practice K is an average and the count of vertices in each Γ_i varies as a function of the sampling density on the mesh surface. In our experiments the ratio K_{max}/K of max to average neighborhood vertex count lies between 2 and 4. We optimize the memory management of our patch operator coefficients $c_{i,j,m} \overset{\text{def}}{=} c_m(\mathbf{u})$ in two ways. First, we reduce their memory footprint from NKM to $4NK$ values by restricting the contribution of each feature at V_j to the 4 neighboring points on the convolution kernel grid, as explained in section 3.2 (see figure 2). Second, to avoid wasting space through padding because of the varying vertex neighborhood size, we map patch extraction coefficients to 1D tensors of $4N_{pairs}$ values, where N_{pairs} is the total number of (V_i, V_j) pairs on the mesh. Each tensor element represents one of the 4 patch extraction coefficients for one such pair. As illustrated on figure 3, the input feature values f_j^d are mapped to this representation using gather operations, and after multiplying the features elementwise with the $c_{i,j,m}$ the result is scattered to an (N, M, D) tensor. Note that the scatter operation accumulates the contributions of the (V_i, V_j) pairs at each patch location m and consequently performs the summation over j indices.

The computation of the convolution as defined by equation (2) involves a 3-level summation over tensors $\{c_{ijm}\}$ of shape (N, K, M), $\{w_m^{d,e}\}$ of shape (M, D, E) and $\{f_j^d\}$ of shape (N, D). In practice, the values of the f_j^d have to be duplicated for each Γ_i to compute the convolution, resulting in a tensor of shape (N, K, D). The 3 reduction operations in (2) must be performed one after the other[2], generating intermediate tensors among which the output of the first reduction is the largest. Thus, the ordering of the summations impacts memory usage. There are three options:

- over j first: $\sum_{j\in\Gamma_i} c_{i,j,m} f_j^d$ has shape (N, M, D)
- over m first: $\sum_{m=1}^{M} c_{i,j,m} w_m^{d,e}$ has shape (N, K, D, E)
- over d first: $\sum_{d=1}^{D} w_m^{d,e} f_j^d$ has shape (N, K, M, E)

Summing over j first allows considerable memory savings as it outputs an order 3 tensor while the other options generate intermediate tensors of order 4. A numerical evaluation on our typical use case illustrates how important the summation order is:

- j first: $N \times M \times D \approx 61.4\,10^6$, or 246 MB
- m first: $N \times K \times D \times E \approx 147\,10^9$, or 590 GB
- d first: $N \times K \times M \times E \approx 36.9\,10^9$, or 147 GB.

Using today's GPU hardware, any option other than summing over j first precludes the implementation of a large number of convolutional layers in mesh registration networks.

[2] Deep learning frameworks offer the possibility of combining several reduction operations using an Einstein summation, but experiments on TensorFlow and PyTorch show that the underlying implementations compute the sums sequentially without optimizing their order for memory consumption.

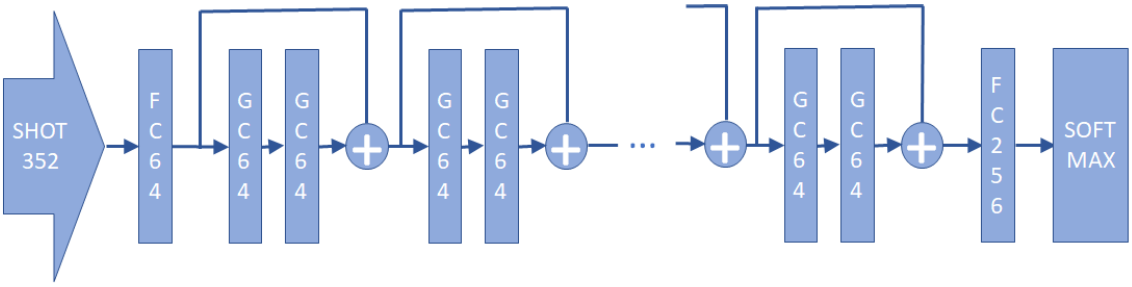


Figure 4: *architecture of our single-resolution deep mesh registration network.*

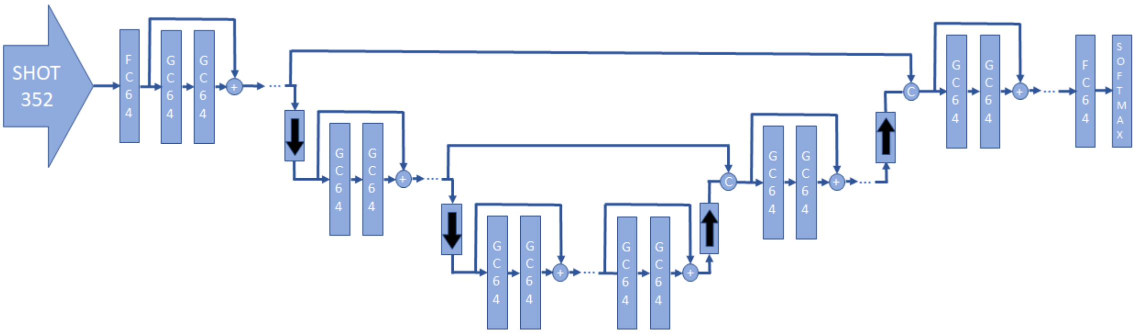


Figure 5: *architecture of our multi-resolution deep mesh registration network. Arrow blocks represent pooling and unpooling layers, C blocks concatenation.*

3.4 Baseline network architecture

The architecture of our baseline network is shown on figure 4. The network is fed with minibatches of SHOT descriptors [Sal14a] for all the vertices of the processed mesh. The registration task is cast as a labeling problem: for each source mesh vertex, the network outputs the probabilities of assignment to each target mesh vertex, and its correpondence is computed as the *argmax* of this vector. The network core consists of memory-optimized geodesic convolutional layers, as described above, each of depth 64. Each pair is set-up as a residual block [HeK16a] by-passed by a skip connection. The number of such blocks is maximized to fit into the available hardware memory.

3.5 Multi-resolution network architecture

In a spatial graph-CNN each vertex is processed independently from its neighbors at inference time. A multi-resolution network as proposed in [Ver18a] introduces some amount of spatial regularization on the outputs, and was shown to improve mesh registration performance on the FAUST dataset. We base our multi-resolution network on the same U-Net architecture but replace the convolutional layers by residual blocks of our memory-optimized layers (figure 5), resulting in a deeper netwrk.

The pooling approach in [Ver18a] is generic for all types of graphs. Following [Ran18a], we specialize it to benefit from the spatial vertex layout information available on a mesh. Pooling in our multi-resolution network relies on the edge collapsing scheme of [Gar97a]. Collapsing an edge maps its two end vertices to one vertex in the decimated mesh, thereby halving the number of vertices. We choose as decimated vertex the edge end that has the smaller collapsing cost.

A pooling layer in our network generates a feature value for each decimated mesh vertex by copying the feature value of the corresponding vertex in the original mesh. Letting N_{orig} be the vertex count of the original mesh, a pooling layer is implemented as a fixed sparse $N_{orig}/2 \times N_{orig}$ matrix that is precomputed from the mesh decimation results. Each row has a unique non-zero value equal to 1 at the location of the collapsed vertex index in the original mesh.

In the same way, the unpooling layer is implemented as a precomputed sparse $N_{orig} \times N_{orig}/2$ matrix. Each row represents a vertex in the upsampled mesh and holds a unique non-zero value equal to 1 at the location of the corresponding vertex in the decimated mesh, computed as the closest vertex.

4 EXPERIMENTAL DATASETS

We validate our approach on two complementary datasets with very different features.

4.1 FAUST [Bog14a]

We experiment with the low-resolution version of the MPI FAUST dataset, built from 100 human body scans with varying poses and morphology. The meshes have 6890 vertices each and are completely free of geometrical artefacts. All meshes have been pre-registered using

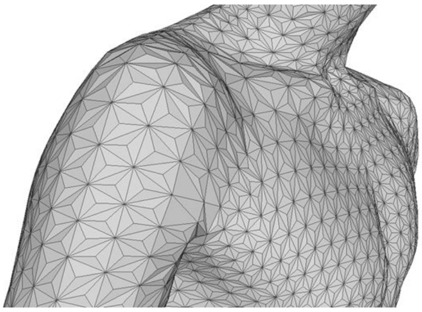


Figure 6: *example FAUST mesh after up-sampling to 15K vertices (detail)*

landmark annotations on the bodies to provide accurate ground-truth correspondence. We follow the protocol of [Mon17a] and split the dataset into 80 meshes for training and 20 for testing. The first mesh in the training dataset is used as the registration target.

To experiment with memory limitations, we generate a higher-resolution version FAUST-15K of the dataset by upsampling the meshes to 15000 vertices (Figure 6). To this purpose we fit a cubic surface patch around each vertex following the algorithm of [Gol04a] and interpolate on this patch the locations of the centroids on each triangle adjacent to the vertex. The centroid for each triangle is computed as the average of these locations. We add centroids to a reference original mesh in the FAUST dataset up to the desired target vertex count, in decreasing order of triangle area. We then select the same set of centroids for all other meshes in order to preserve the ground-truth per-vertex correspondence of the original dataset.

4.2 Bosphorus [Sav08a]

We also validate our approach on Bosphorus, a dataset of higher-resolution, noisy expressive facial scans that is more representative of the 3D data processed in the industry. The scans come as point clouds with holes and sometimes noticeable artifacts, which we pre-process as shown on Figure 7. After removing background points by depth thresholding, the point clouds are meshed using 2D Delaunay triangulation in a plane perpendicular to the scanning axis (Figure 7b). Next, we fill holes and remove small connected components (Figure 7c). Finally, we apply quadric edge decimation to downsample all meshes to 15K vertices (Figure 7d). The target mesh is set to a neutral expression scan that belongs neither to the training set nor to the test set, and is pre-processed in the same way. Ground truth correspondences to this target are obtained using the deformation transfer algorithm of [Sum04a], leveraging the landmark annotations provided with the Bosphorus dataset. We validate the excellent registration quality of this method through high-resolution texture transfer. 150 scans are selected for training and 50 for testing.

5 EXPERIMENTAL RESULTS

5.1 Methodology

We assess the performance of our registration approach using the Princeton benchmark protocol [Kim11a], and compare it with two recent approaches [Mon17a, Ver18a] for which the authors provide their code. The registration accuracy is plotted as the proportion of correct correspondences over the test set (vertical axis) as a function of the tolerance on the geodesic error (horizontal axis), expressed as a ratio of the mesh diameter. To allow for a fair comparison we optimize the network and training parameters separately for all approaches given a fixed GPU memory budget. We run all our experiments on an NVidia Tesla P100 GPU with 16 GB memory. We use as network inputs for all datasets 352-dimensional SHOT descriptors [Sal14a]. Minibatches consist of the descriptors for all the vertices of a mesh, except for [Mon17a] where they are restricted to mesh regions to avoid running out of memory. In this case the processing of sub-meshes is managed in the authors code.

Besides the number of convolutional layers in the network, the parameters most impacting the memory requirements are the geometrical extent ρ_{max} of the vertex neighborhoods and the size M of the convolution kernel. In our approach, kernels are sampled on 8 by 8 regular grids in the local geodesic polar systems around each vertex, hence M is set to 64. For [Mon17a, Ver18a] we use the values of M advocated in the papers. For each approach we optimize the remaining parameters by performing a grid search, selecting the values that yield the best registration accuracy while not running out of memory. These parameters are ρ_{max} for [Ver18a], ρ_{max} and the count of vertices in each minibatch for [Mon17a], and for our approach ρ_{max} and the number of convolutional layers.

The network is trained by minimizing the standard cross-entropy loss by means of an Adam optimizer, with an initial learning rate of 10^{-4} and over 800 epochs. We use batch normalization but no dropout.

5.2 Single-resolution performance

Figure 8 shows the compared accuracy of our approach against MoNet [Mon17a] and the single resolution version of FeaStNet [Ver18a]. For completeness we also plot the performance curves of older shape matching approaches on FAUST, reproduced from [Lit17a] and [Mas15a].

Our scheme outperforms all previous single-resolution approaches on the 3 considered datasets. Unsurprisingly, the accuracy on Bosphorus for all methods is considerably less than on FAUST, a consequence of the large amount of geometrical noise of this dataset. However, for the same quality of geometry (FAUST and

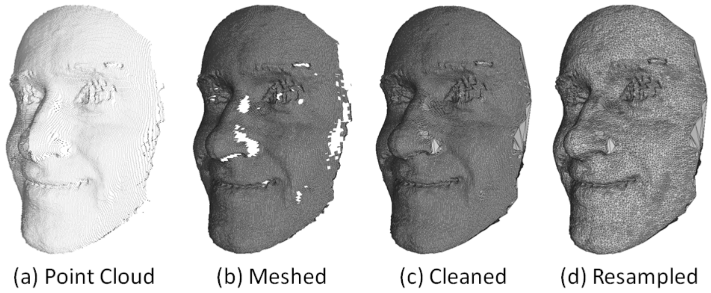


Figure 7: *Pre-processing pipeline of the Bosphorus dataset point clouds*

Algo - Database	Num conv. layers	ρ_{max}	M	Batch size
[Mon17a] - FAUST	3	0.60 IOD	80	75 vertices
[Mon17a] - FAUST 15K	3	0.30 IOD	80	150 vertices
[Mon17a] - Bosphorus	3	0.06 IOD	80	150 vertices
[Ver18a] - FAUST	3	1-ring	32	1 mesh
[Ver18a] - FAUST 15K	3	1-ring	32	1 mesh
[Ver18a] - Bosphorus	3	1-ring	32	1 mesh
Ours single-res - FAUST	30	1.20 IOD	64	1 mesh
Ours single-res - FAUST 15K	30	1.20 IOD	64	1 mesh
Ours single-res - Bosphorus	14	0.25 IOD	64	1 mesh
Ours 2 res levels - FAUST	10+10+10	1.20 IOD	64	1 mesh
Ours 2 res levels - FAUST 15K	10+10+10	0.90 IOD	64	1 mesh
Ours 2 res levels - Bosphorus	10+10+10	0.18 IOD	64	1 mesh

Table 1: *Optimal experimental settings of memory-dimensioning parameters (IOD: Inter-Ocular Distance)*

FAUST-15K), our approach maintains its performance level when the vertex count is increased, while the accuracy of [Mon17a] and [Ver18a] drops significantly.

Figure 9 a) and b) demonstrates the improvement of the registration accuracy as more convolutional layers are added to the network. On Bosphorus performance keeps improving up to the depth limit set by the hardware memory capacity.

5.3 Multi-resolution performance

Table 2 summarizes the performance of our multi-resolution network on the FAUST and FAUST-15K datasets. Our architecture has 2 resolution levels and 10 convolutional layers in each of the sections delimited by the pooling and unpooling layers. Adding more layers again improves performance, but the gain becomes less significant as our registration scores approach perfection. On FAUST our performance is superior to the results reported in [Ver18a], the only point of comparison in this case.

On Bosphorus the multi-resolution accuracy improves with network depth but remains lower than using a single-resolution network (Figure 9 c). We hypothesize that the noisier geometry on this dataset hinders the convergence of the network towards stable vertex correspondence patterns and thereby cancels the benefits of the multi-resolution analysis of the meshes.

Database	[Ver18a]	Ours(10+10+10)
FAUST	0.9860	**0.99997**
FAUST 15K	N/A	**0.9984**

Table 2: *multi-resolution registration accuracy at zero geodesic error for [Ver18a] and the optimal configuration of our multi-resolution network)*

5.4 Qualitative experimental results

Figure 11 shows qualitative registration results on samples from the FAUST and Bosphorus datasets. Geodesic registration errors are translated into cm assuming an interocular distance of 6.3 cm.

On FAUST most of the registration errors are localized on the torso, which lacks distinctive geometrical features for shape matching. For the same reason, the largest registration errors on Bosphorus occur on the peripheral areas of the meshes, away from the main facial features. The approach of [Ver18a] tends to map neighboring source vertices to the same target location, creating colocated vertices that give the morphed mesh a lower-resolution appearance.

The morphs of the source meshes to the geometry of the target mesh on Figure 11 should be interpreted with care. Most conspicuous in these morphs are the large

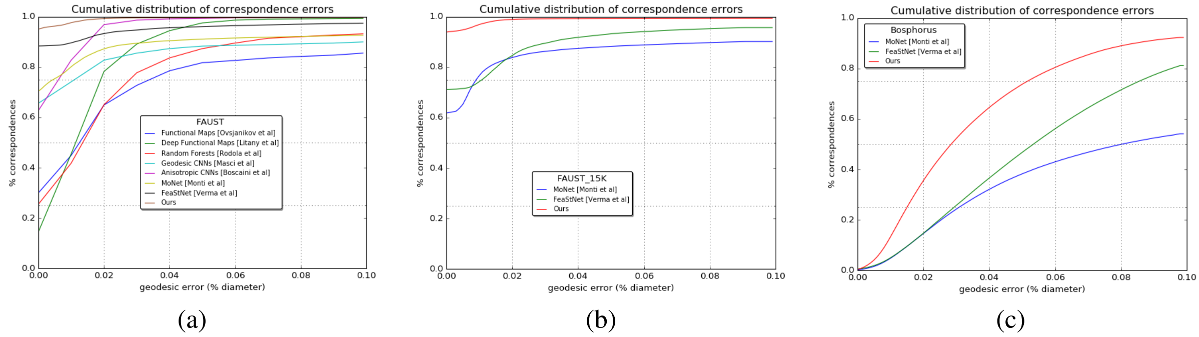


Figure 8: *registration accuracy benchmark on FAUST (a), FAUST 15K (b) and Bosphorus (c) datasets*

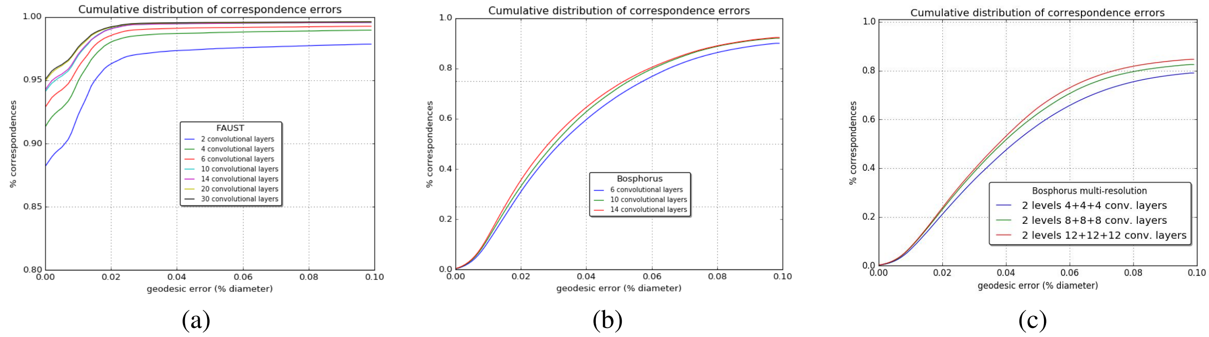


Figure 9: *influence of the number of convolutional layers on registration accuracy (a) Faust single-resolution, (b) Bosphorus single-resolution, (c) Bosphorus multi-resolution*

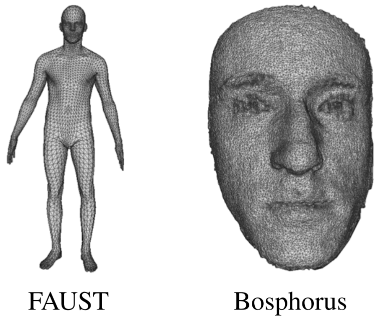


Figure 10: *FAUST and Bosphorus registration targets.*

triangles resulting from often isolated wrong vertex correspondences. It should be borne in mind that such triangles account only for a small proportion of the registration errors, as evidenced by the color encoding representations.

6 CONCLUSION

When processing 3D meshes using deep networks, increased memory requirements resulting from the generalization of convolution operators to non-Euclidean domains set a limit to the number of convolutional layers that can be placed in the network. The memory-friendly convolution scheme we propose mitigates this issue and allows deeper networks to be implemented for a given memory budget. On the mesh registration task we showed that deeper network architectures result in substantial performance gains.

Using a multi-resolution network featuring pooling and unpooling operators specialized for 3D mesh processing, we obtain close to perfect registration scores on the FAUST dataset. However, on noisy Bosphorus meshes the registration quality leaves much room for improvement. We believe FAUST is a toy dataset that has been useful for benchmarking mesh registration results so far, but should now been abandoned in favor of collections of meshes that are more representative of the noisy and higher-resolution scans used in the industry. In this respect, Bosphorus is arguably a worst case. The progress of 3D capture technology and photogrammetry in recent years should allow the constitution of datasets of better geometrical quality.

Further issues remain with the proposed registration scheme. First, like all competing approaches it takes as inputs meshes with different topologies and registers them to a common template. In practical use cases the reverse correspondence is needed to transfer the reference topology in a source mesh to a set of captured scans. Second, the local intrinsic mapping of vertex neighborhoods on the mesh surface to Euclidean convolution patches incurs heavy pre-processing as geodesic paths need to be computed in every vertex neighborhood. Simplifying this mapping would be desirable in order to better benefit from the efficiency of deep learning inference on GPU hardware.

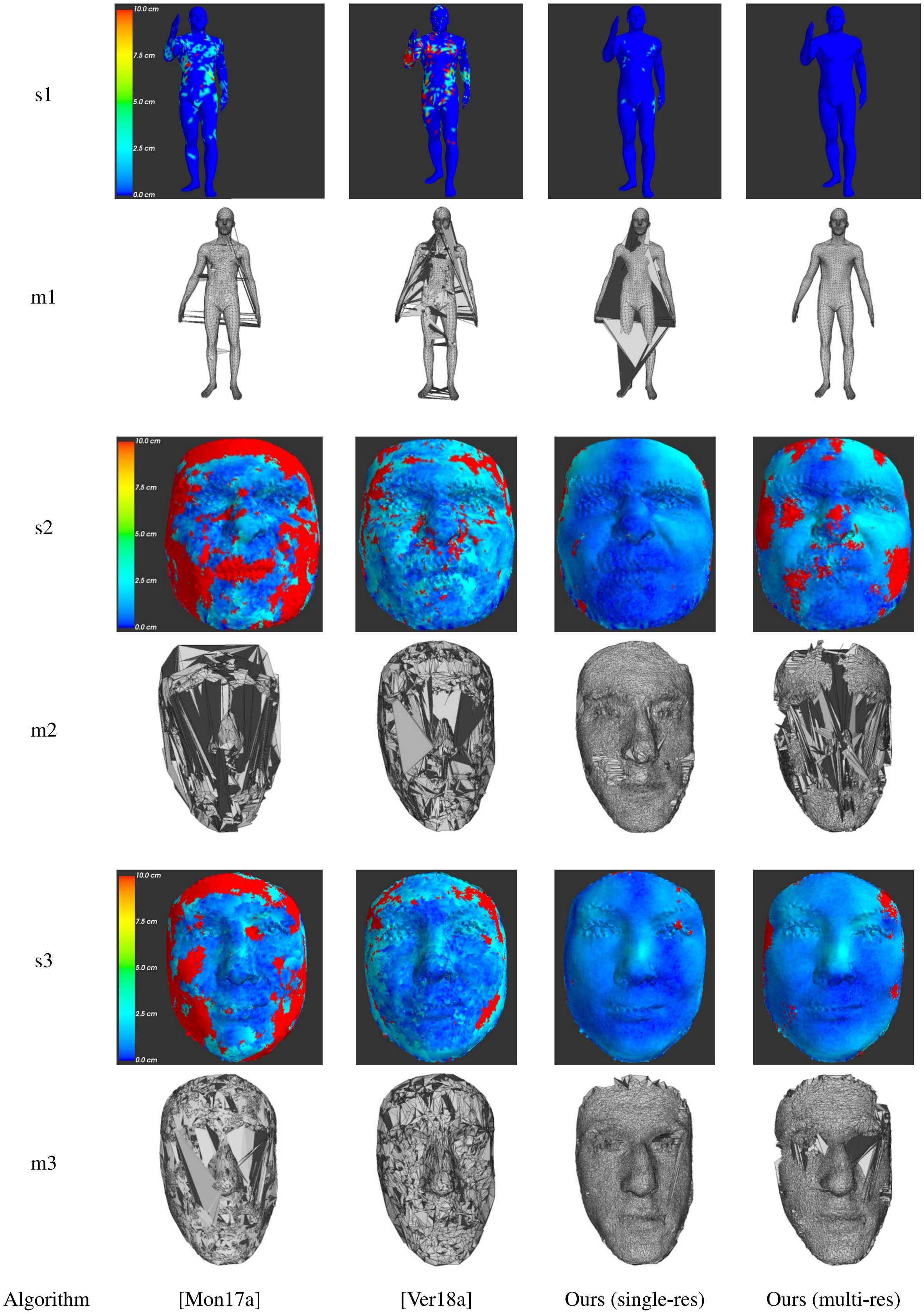


Figure 11: *per-vertex registration errors for FAUST (s1) and Bosphorus (s2, s3) samples. Rows s1, s2, s3 display a color encoding of the geodesic correspondence errors on the source meshes. Rows m1, m2, m3 show the morphs of the source meshes to the target, based on correspondence results. Best viewed in color.*

7 REFERENCES

[Amb07a] Amberg, B., Romdhani, S. and Vetter, T., Optimal Step Nonrigid ICP Algorithms for Surface Registration, in Conf. Proc. CVPR'07, 2007.

[Bog14a] Bogo, F., Romero, J., Loper, M., and Black, M.; FAUST: dataset and evaluation for 3D mesh registration, in Conf. Proc. CVPR'14, 2014.

[Bos16a] Boscaini, D., Masci, J., Rodola, E., and Bronstein, M., Learning shape correspondence with anisotropic convolutional neural networks, in Conf. Proc. NIPS'16, 2016.

[Bro17a] Bronstein, M., Bruna, J., LeCun, Y., Szlam, A., and Vandergheynst, P., Geometric deep learning: going beyond Euclidean data, IEEE Sig. Proc. Magazine, vol. 34, no. 4, pp. 18-42, 2017.

[Gar97a] Garland, M., and Heckbert, P.S., Surface Simplification Using Quadric Error Metrics, in Conf. Proc. SIGGRAPH'97, 1997.

[Gol04a] Goldfeather, J., and Interrante, V., A Novel Cubic-Order Algorithm for Approximating Principal Direction Vectors, ACM Trans. on Graphics Vol.23, No.1, pp.45-63, 2004.

[HeK16a] He, K., Zhang, X., Ren, S., and Sun, J., Deep Residual Learning for Image Recognition, in Conf. Proc. CVPR'16, 2016.

[Kai04a] van Kaick, O., Zhang, H., Hamarneh G., and Cohen-Or D., A Survey on Shape Correspondence, Computer Graphics Forum, Vol.30, No.6, p.1681-1707, 2011.

[Kim11a] Kim, V., Lipman, Y., and Funkhouser, T., Blended Intrinsic Maps, ACM Trans. on Graphics Vol.30, No.4, pp.79:1-79:12, 2011.

[Kip17a] Kipf, T., and Welling, M., Semi-Supervised Classification with Graph Convolution Networks, in Conf. Proc. ICLR'17, 2017.

[Lit17a] Litany, O., Remez, T., Rodola, T. and Bronstein, A., Deep Functional Maps: Structured Prediction for Dense Shape Correspondence, in Conf. Proc. ICCV'17, 2017.

[Mas15a] Masci, J., Boscaini, D., Bronstein, M., and Vandergheynst, P., Geodesic Convolutional Neural Networks on Riemaniann Manifolds, in Conf. Proc. ICCV'15 Workshop, 2015.

[Mon17a] Monti, P., Boscaini, D., Masci, J., Rodola, E., Svoboda, J., and Bronstein, M., Geometric deep learning on graphs and manifolds using mixture model CNNs, in Conf. Proc. CVPR'17, 2017.

[Ovs12a] Ovsjanikov, M., Ben-Chen, M., Solomon, J., Butscher, A. and Guibas, L., Functional Maps: a Flexible Representation of Maps between Shapes, ACM Trans. on Graphics, Vol.31, No.4, 2012.

[Ran18a] Ranjan, A., Bolkart, T., Sanyal, S., and Black, M.J., Generating 3D Faces using Convolutional Mesh Autoencoders, in Conf. Proc. ECCV'18, 2018.

[Rod14a] Rodolà, E., Bulo, S., Windheuser, T., Vestner, M. and Cremers, D., Dense Non-rigid Shape Correspondence using Random Forests, in Conf. Proc. CVPR'14, 2014.

[Sal14a] Salti, S., Tombari, F., and Di Stefano, L., SHOT: Unique Signatures of Histograms for Surface and Texture Description, Comp. Vis. and Im. Underst., Vol 125, pp.251-264, 2014.

[Sav08a] Savran, N., Alyüz, N., Dibeklioglu, H., Celiktutan, O., Gökberk, B., Sankur, B., and Akarun, L., Bosphorus Database for 3D Face Analysis, in Conf. Proc. COST 2101 Workshop on Biometrics and Identity Management (BIOID 2008), 2008.

[Sum04a] Sumner, R., and Popovic, J., Deformation Transfer for Triangle Meshes, ACM Trans. on Graphics Vol.23, No.3, pp.339-405, 2004.

[Ver18a] Verma, N., Boyer, E., and Verbeek, J., FeaSt-Net: Feature-Steered Graph Convolutions for 3D Shape Analysis, in Conf. Proc. CVPR'18, 2018.

[Wei16a] Wei, L., Huang, Q., Ceylan, D., Vouga E., and Li, H., Dense Human Body Correspondences using Convolutional Networks, in Conf. Proc. CVPR'16, 2016.

[Zel13a] Zell E., and Botsch, M., ElastiFace: Matching and Blending Texture Faces, in Conf. Proc. ACM Symposium on Non-Photorealistic Animation and Rendering, 2013.

ISSN 2464-4617 (print)
ISSN 2464-4625 (DVD)

Computer Science Research Notes
CSRN 3001

WSCG2020 Proceedings

Coarse Classification of Teeth using Shape Descriptors

Katarzyna Gościewska and Dariusz Frejlichowski
West Pomeranian University of Technology, Szczecin
Faculty of Computer Science and Information Technology
Żołnierska 52, 71-210, Szczecin, Poland
{kgosciewska,dfrejlichowski}@wi.zut.edu.pl

ABSTRACT

This paper presents the problem of coarse classification in an application to teeth shapes. Coarse classification allows to separate a set of objects into several general classes and can precede more detailed identification or narrow the search space. Features of an object are mainly determined by its geometrical aspects, therefore we investigate the use of shape description algorithms, namely the Two-Dimensional Fourier Descriptor, UNL-Fourier Descriptor, Generic Fourier Descriptor, Curvature Scale Space, Zernike Moments and Point Distance Histogram. During the experiments we examine the accuracy of classification into two classes: single-rooted teeth and multi-rooted teeth—each class has five representatives. We also employ an additional step of data reduction. Reduced representations are obtained in three ways: by taking a part of an original representation, by predefining a shape description algorithm parameter or by applying an additional step of data reduction technique, i.e. the Principal Component Analysis or Linear Discriminant Analysis. Euclidean distance is used to match final feature vectors with class representatives in order to indicate the most similar one. The experimental results proved the effectiveness of the proposed approach.

Keywords

teeth separation, coarse classification, shape descriptors, data reduction, dental radiographs

1 INTRODUCTION

The application of teeth as a biometric feature is accepted worldwide and appreciated especially in the field of forensic odontology, which is the science of dentistry related to law. Various forms of dental evidence are used in the identification process, such as entire dentitions, tooth fragments, bite mark impressions, dental treatment histories, including dental radiographs, dentition anomalies and dental works. Full permanent dentition consists of 32 teeth divided into four groups: 8 incisors, 4 canines, 8 premolars and 12 molars. The size of upper and lower teeth of the same type varies. Upper incisors are bigger than lower incisors, while upper molars are smaller than lower molars. All first molars are larger than second and third molars, and third molars are the smallest molars in the mouth, but generally molars are the largest teeth of the permanent set of teeth. All incisors, canines and premolars are single-rooted, while lower molars are double-rooted, and upper molars are triple-rooted. Upper molar roots are more or less fused [Gra00].

Radiographic dental images are extensively used in the analysis and identification of human, and X-ray imaging enables to obtain a considerable amount of data. Various types of radiographs can be obtained (depending on the view and the part of the mouth that is being imaged), however not all of them are equally often utilized in the identification process [Vin08]. Three types of dental radiographs—periapical, bitewing and panoramic radiographs—are presented in Figure 1. Periapical radiography produces intraoral radiograph images that most frequently depict three or four teeth with surrounding tissue and are mainly used in diagnosis. A bitewing radiograph is used to depict some of the teeth in left or right side of the jaws, namely molars, premolars and canines [Che08]. The third type of radiograph is an orthopantomogram, which is a panoramic, two-dimensional view of the full dentition, both jawbones and supporting structures from ear to ear.

The major sources of dental features are bitewing and panoramic radiographs. The most distinguishable substances visible on dental radiographs are dental

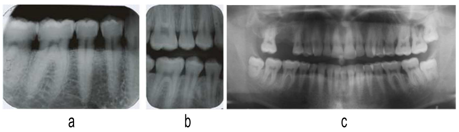


Figure 1: Sample radiographs: a) periapical [Jai05], b) bitewing [Che09], c) panoramic [Jai05].

https://doi.org/10.24132/CSRN.2020.3001.2

fillings, especially amalgam restorations [Phi09]. Although dental restorations pay a significant role in the identification process, due to improved dental care and minimal restorations (modern filling materials have poor radiographic characteristics), other oral features are assessed during the identification [Pre01], such as the shape of teeth, both crown and root, and teeth appearance (grey level). This reason, coupled with the significant amount of dental records and low efficiency of manual methods, has led to the development of automatic identification techniques, and ultimately to the creation of automated dental identification systems. Given an input image, usually a postmortem radiograph, a search is performed in order to find the best matching antemortem radiograph in the database. This problem can be therefore considered as an image matching and retrieval problem [Mar11]. Moreover, the use of dental biometrics, compared to other biometric visual features (such as fingerprints [Yan10] or ear shape [Sul15]), can be performed regardless of the condition of soft tissues.

An automated dental identification system consists of three main steps: feature extraction, atlas registration and matching of dental radiographs [Che09]. Teeth classification is an important step preceding teeth numbering and subsequent matching, and its importance stems from several reasons. Firstly, the quality of classification affects subsequent processing steps. Secondly, due to the diversity of human dentition, particularly in molars appearance—the shape of crowns and roots and the number of the roots—person identification can be limited solely to the use of molar features. This approach also narrows the search space and reduces the computation time at matching stage. Bitewing radiographs are usually used for teeth classification into molars and premolars, and the results of the classification are then verified to check if they follow specific patterns. However, bitewing projection provides only partial information about an individual dentition and panoramic radiographs should be considered, particularly in the classification of teeth into four classes (molars, premolars, canines and incisors, eg. [Nas08]) or three classes (e.g. [Jai05]).

This paper considers and examines the problem of tooth shape classification using various approaches based on shape description algorithms. Panoramic radiographs are used as input images for tooth contour extraction. Contours are then represented using shape descriptors and divided into two classes—molars (multi-rooted) and non-molars (single-rooted) based on the similarity to the previously prepared templates. It is desired to find the best solution for this classification task, combining the highest percentage of classification accuracy and the smallest size of shape description.

The remaining part of the paper is organised as follows: the second section describes some related works on representation and classification of tooth contours. The third section contains the description of the proposed approach and presents algorithms selected for the experiments. The fourth section presents experimental conditions and results, and the last section summarizes and concludes the paper.

2 RELATED WORKS

This section presents some works concerning methods which are used for shape representation/feature extraction and teeth classification. We are mostly interested in applications of various shape description algorithms to teeth silhouettes. The second area of interest is a matching process performed during classification stage. Moreover, we intent to find an appropriate solution for obtaining small and compact representations. Several approaches that meet some of these requirements are presented below.

In [Mah05], Mahoor and Abdel-Mottaleb provided the solution for teeth classification and numbering in bitewing radiographs. Teeth are classified using the Bayesian classification into molars and premolars and later an absolute number is assigned to each tooth according to the common numbering system. The approach is based on tooth contours and two different kinds of Fourier descriptors are used as features for the classification—complex coordinates signatures and centroid distance of the contours. The arrangement of the teeth is taken into consideration in order to correct any misclassifications and to perform teeth numbering.

In [Bar12], Barboza et al. proposed the use of two different shape descriptors as biometric features for human identification. The authors used a graph-based algorithm for tooth contours extraction from panoramic radiographs. Tooth contours were represented using the Shape Context and Beam Angle Statistics (BAS) descriptors. Slightly better results were obtained for the matching of BAS representations. The majority of failures were attributed to the radiographs with poor segmentation.

Raju and Modi [Raj11] introduced a novel approach to feature extraction based on the multiple features of tooth shape and texture. The shape analysis is performed using Fourier Descriptors, and the texture analysis utilizes Grey Level Co-occurrence Matrix and its various properties such as Energy, Contrast, Correlation and Homogeneity. For feature matching the mean square error is calculated between the query and database radiographs.

Pattanachai [Pat12] proposed the use of Hu's moment invariants as tooth features and the Euclidean distance for feature matching. Another moment-based approach is described in [Gho12]. Ghodsi and Faez proposed the Zernike Moments for shape description in two steps: high-level features were used to reduce search space and low-level features were matched using the Euclidean distance.

In [Kuo10], Kuo and Lin presented a method for dental work extraction from bitewing radiographs which comprises two stages: the location of the coarse contours of all dental works and the utilization of region growing technique to obtain complete dental works. The matching approach uses two metrics: frequency domain based on Fourier Descriptors and spatial domain based on the relative size of the misaligned region between two matched dental works.

Nassar et al. [Nas08] proposed a two-stage approach to the automatic classification of teeth into four classes, i.e. molars, premolars, canines and incisors. In the first stage, some appearance-based features are used to assign initial classes, while in the second stage a string matching technique is used for class validation and assigning of tooth numbers. The method used in the second stage is based on teeth neighbourhood rules. The classification approach is applied for periapical and bitewing radiographs, achieving an accuracy rate of 87%.

Arifin et al. [Ari12] proposed a novel method for classification of teeth into molars and premolars on bitewing radiographs. The approach utilizes a support vector machine for classification and mesiodistal neck detection for feature extraction.

In [Als12] Al-sherif et al. proposed the utilization of appearance-based Orthogonal Locality Preserving Projection algorithm for assigning initial classes to teeth on bitewing radiographs. Later, a string matching technique is used to validate initial classes and finally to assign tooth numbers. The proposed approach achieved classification accuracy of 89%, which was enhanced by class validation to the overall accuracy of 92%.

Yuniarti et al. [Yun12] proposed a system for human identification, which utilizes the binary SVM method for teeth classification into molars and premolars using three tooth features: area, a ratio of height to width and centroid. Next, the numbering is applied to avoid incorrect teeth patterns. The accuracy of the SVM classification amounted to 89.07% and was subsequently improved to 91.6% by pattern correction.

3 THE PROPOSED APPROACH

In the paper, an approach based on the classification of teeth extracted from panoramic radiographs into single-rooted and multi-rooted teeth classes is proposed. The main goal of this classification is to narrow the number of teeth used for identification. Due to the fact that shape of molars is more diversified and varies visibly from (bi)cuspids and incisors, high classification accuracy values are expected. Panoramic radiographs are less frequently utilized for human identification due to uneven illumination and magnification as well as teeth occlusion they are more difficult to process. Nevertheless, orthopantomograms are still a valuable source of dental data, because they illustrate teeth with crowns and roots, their relative positions in the mouth and surrounding structures on a single image.

All tooth shapes were obtained from panoramic radiographs using the approaches proposed by Frejlichowski and Wanat in [Fre10b, Fre11a, Fre11b]. Three stages were involved in the preparation of tooth shapes for experimental databases: image enhancement, radiograph segmentation, and extraction of tooth contours. The enhancement of image quality was performed using the Laplacian Pyramid Decomposition. As a result, images had improved contrast, sharper edges and the difference between teeth and surrounding bones was more visible [Fre10b]. Image segmentation was performed on the basis of the locations of areas between necks of teeth, which were used for determining separating lines [Fre11a]. For tooth contour extraction a novel method was utilized. An image was segmented using the watershed algorithm and resulting regions were classified as belonging to the tooth or to the background by means of a fitness function. Regions, considered as belonging to the tooth, had their pixels set to 1, whereas other regions were rejected. Afterwards, the remaining regions were processed by means of dilation and traced to find external boundaries. Tooth contours were smoothed using Gaussian filtering [Fre11b]. The resulting list of points for each tooth was plotted on the image plane and saved.

In the next step, each tooth contour is represented using selected shape descriptor—six various description algorithms were chosen, namely the Two-Dimensional Fourier Descriptor (2DFD) [Kuk98], Generic Fourier Descriptor (GFD) [Zha02], UNL-Fourier Descriptor (UNL-F) [Rau94], Curvature Scale Space (CSS) [Abb99] with an additional Fourier Transform step, Zernike Moments (ZM) [Yan08] and Point Distance Histogram (PDH) [Fre10a]. In the proposed approach, the classification is based on feature matching using the Euclidean distance. Each of the two classes is represented by five binary tooth shape images, i.e. templates. The database consists of test objects, i.e. binary tooth shape images extracted from panoramic radiographs. Properly prepared description vectors of test objects are matched with template description vectors—the nearest template indicates the class of the test object. A brief description of applied algorithms is provided below.

Owing to its useful properties, the Fourier Transform is widely used in pattern recognition. The Two-Dimensional Fourier Descriptor applies Fourier Transform to a region shape (a contour with its interior) and the resultant representation has the form of a matrix with absolute complex values [Kuk98]. The UNL-Fourier descriptor is composed of the UNL (named after Universidade Nova de Lisboa) descriptor and Two-Dimensional Fourier Transform. The use of the UNL results in a Cartesian image containing the unfolded shape contour as it is seen in polar coordinates—the rows represent distances from the centroid, and the columns the corresponding angles. Then the Two-Dimensional Fourier Descriptor is applied to obtain the UNL-F representation. The Generic Fourier Descriptor is a region-based Fourier Descriptor that utilizes the transformation to the polar coordinate system. All pixel coordinates of an original region shape image are transformed into polar coordinates and new values are put into a rectangular Cartesian image [Zha02]. The row elements correspond to distances from the centroid and the columns to corresponding angles. As a result, an image of a transformed shape is obtained and the Two-Dimensional Fourier Transform can be applied.

The Curvature Scale Space is a contour shape descriptor based on multi-scale representation and curvature. CSS representation is obtained by tracking zero-crossing points of the curve while it is iteratively smoothed by Gaussian function. At each level, as the Gaussian kernel width increases, the curve becomes smoother and the number of zero crossing points on the curve decreases. The generation of subsequent, smoother curves is called an evolution. If the locations of the curvature crossing points are known, the results can be displayed on the image plane called a CSS image. The column elements of the CSS image refer to the representative contour points and row elements corresponds to the Gaussian kernel widths [Abb99]. Instead of extracting the maxima of the CSS contours, the CSS image is represented as a binary image, and the Two-Dimensional Fourier Transform is applied as an additional step.

The Zernike Moments are orthogonal moments which can be derived using Zernike orthogonal polynomials. The Zernike polynomials are a complete set of functions orthogonal over the unit disk $x^2 + y^2 < 1$. The Zernike Moments are rotation invariant and resistant to noise and minor variations in shape [Yan08].

The Point Distance Histogram is a contour-based shape descriptor which utilizes the transformation of contour points from Cartesian to polar coordinates. As a result, the representation of a shape is invariant to translation and scaling provided that normalization is applied. In order to obtain basic shape representation, the centroid is calculated. Next, the shape contour is transformed into polar coordinates, and new coordinates are put into two vectors—Θ^i for angles and P^i for radii. Values in Θ^i are converted to the nearest integers. In the next step, the elements of Θ^i and P^i are sorted according to increasing values in Θ^i and denoted as Θ^j and P^j. If there are any equal angle values in Θ^j, only the value with the highest corresponding radii value in P^j is left. These transformations produce a vector consisting of no more than 360 elements, and only P^j is further processed (denoted as P^k). The P^k vector is normalized according to its highest value. The elements in P^k are assigned to bins in a histogram (ρ_k to l_k). In the next step, the values in bins are normalized according to the highest one and final histogram is obtained [Fre10a].

The Principal Component Analysis is an unsupervised, linear dimensionality reduction technique. It enables the construction of low-dimensional representation of the data, which describes the most variability of the original data. Dimensionality reduction is obtained by finding a linear combinations of the original variables, which are uncorrelated and are characterised by the highest variance. These combinations are called principal components. For instance, the second component is linearly combined with the second highest variance value and is orthogonal to the first component. In many cases, a small number of first components reflects the highest variability of the data. The remaining components are deleted—although this results in data reduction, the loss of information is small [Fod02]. In the proposed approach, a matrix of feature vectors is used as an input for PCA—one row corresponds to one feature vector. After the PCA is applied, a new set of data is obtained. Then each row corresponds to the reduced feature vector, which contains from 1 to 10 principal components.

The Linear Discriminant Analysis is applied as a supervised data reduction technique. It is used to find a linear combination of features which best explains the data and preserves information about class labels. In other words, the method is focused on finding such data transformation that will maximize the separation between classes and minimalize the separation within classes [Cun07]. In practice, an input matrix is the same as the one used for PCA, and in addition the vector of class labels is given. Moreover, from an algorithmic point of view, LDA utilizes PCA as a step, therefore a various number of principal components can be used in the experiments. However, the final reduced representation includes two LDA components due to the consideration of the two-class classification problem.

The main focus of the proposed approach is to choose shape features that will ensure accurate classification. However, an attempt is made to maximally reduce the size of the shape description vector in order to minimize the storage space and to reduce the matching time. Therefore, various sizes of feature vectors were prepared for the experiments. The first set of experiments included the original shape representations. For the Zernike Moments and Point Distance Histogram the representations were generated using various orders and numbers of histogram bins respectively. For other shape descriptors using the Two-Dimensional Fourier Transform, which produces a coefficient matrix, various subparts of the original matrix were taken into account. In the subsequent experiments an additional data reduction step is performed prior to feature matching and two techniques are utilized—the Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA).

4 EXPERIMENTAL RESULTS

The main goal of the experiments was to choose the best shape description method for teeth classification into multi-rooted and single-rooted classes. The shape representation should be compact, therefore the descriptions were reduced in various ways. The experimental database consisted of 903 tooth contour images, extracted from panoramic radiographs of 30 different persons. Ten other tooth shapes were extracted from a separate set of radiographs and prepared for the template database (see Figure 2). Each class was represented by five template images. Since the original classes were known, it was possible to obtain a percentage accuracy of the classification.

In total, eighteen experiments were performed. During each experiment, various shape descriptions' sizes (or variants) were taken into consideration, as well as the different number of principal components if it was applicable. Firstly, all shapes in the database and the templates were represented by the same variant of the shape descriptor. Secondly, the Euclidean distance between each test object and template was calculated. The template with the smallest dissimilarity value indicated the class of the test object (the closest match). Finally, the classification accuracy as well as the ratio of correctly classified shapes to the number of all known class members was estimated for each class.

The first set of six experiments concerned the utilization of shape descriptors and their various variants, parameters or sizes. The representations obtained using shape description algorithms utilizing the Fourier Transform were manually reduced to smaller sizes by selecting square subparts of the Fourier coefficient matrix. The experiment using the PDH descriptor was performed for various numbers of histogram bins, whereas the experiment that utilized the Zernike Moments was carried out for the moments of various orders. All features used in the first set of experiments are considered as 'original'. The highest percentage accuracy values of each experiment and various description parameters are tabulated in Table 4.

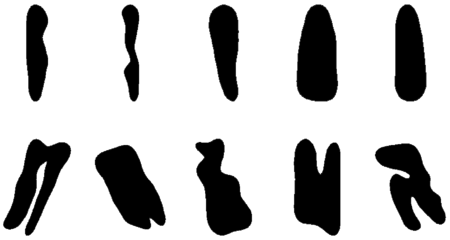


Figure 2: Templates used in the experiments: single-rooted class representatives are shown in the first row, whereas multi-rooted class representatives are presented in the second one.

The best results, exceeding the accuracy of 90%, were obtained for 2DFD, GFD and ZM, however they concerned the classification into single-rooted class. The classification accuracy to multi-rooted class was worse and not satisfactory. The highest accuracy value reached 81% and was observed in the experiment utilizing CSS+2DFD. For this reason, it was assumed that the manual selection of the size of the description vector is inefficient—it was probably caused by the fact that the original shape representation either contains additional information which worsen the results, or the feature vector could not appropriately reflect all distinctive and important shape features. Therefore, two additional sets of the experiments were performed.

In the second set, the Principal Component Analysis step was added. All shapes were represented by the ap-

Shape Descriptor	**Class**	**Accuracy**	**Variant**
2DFD	Multi-rooted	61.5%	2×2 subpart
	Single-rooted	93.7%	15×15 subpart
CSS	Multi-rooted	81.4%	10×10 subpart
	Single-rooted	73.5%	25×25 subpart
GFD	Multi-rooted	61.8%	2×2 subpart
	Single-rooted	93.2%	10×10 subpart
UNL-F	Multi-rooted	68.8%	5×5 subpart
	Single-rooted	86.2%	15×15 subpart
PDH	Multi-rooted	72.8%	2 bins
	Single-rooted	79.4%	50 bins
ZM	Multi-rooted	60.2%	1st order
	Single-rooted	94.0%	8th order

Table 1: The experiments utilizing shape descriptors.

propriate shape description vector in the same way as before, however the smallest shape representation size had to be larger than the largest number of target principal components. Afterwards, all shape representations were reduced to one to ten principal components. Finally, matching was performed on the basis of reduced representations, and the percentage classification accuracy was estimated for the combination of each shape description vector size and each number of principal components. The highest accuracy values of each experiment are provided in the Table 4.

The best classification results were obtained in the experiment using CSS+2DFD. The accuracy values were nearly equal for both classes and amounted to 93.4% for the multi-rooted class, and 95.7% for the single-rooted class. The second rank was scored by the PDH descriptor which achieved accuracy at the level of 91.2% for molars and 92.5% for incisors and (bi)cuspids. The results obtained for the other descriptors are more varied between classes.

In the third set, the experiments were carried out in the same way as before, with the difference that instead of the PCA, the LDA method was used. The experiments were performed for one to ten PCA components. All of the matched feature vectors had two elements after reduction, due to the fact that classification into two classes by means of LDA produces shape description composed of two components. The highest percentage classification accuracy values are tabulated in Table 4.

The experimental results obtained with the use of LDA instead of PCA resulted in a slight improvement. This time the highest accuracy value achieved 96.9% for the experiment utilizing 2DFD and for single-rooted classification. Unfortunately, the application of 2DFD for multi-rooted teeth classification yielded poor results. The best overall effectiveness of the classification to both classes can be attributed to the Point Distance Histogram (95.2% and 92.5%).

Table 4 and Table 4 contain a summarized representation of best results. Each row contains the best percentage accuracy values obtained for a particular shape descriptor in its original form and with the application of additional data reduction step. 'Variant' refers to the sizes or parameters of the feature vectors that were matched during experiments. The results are presented separately for each class.

Taking into consideration the accuracy values together with the sizes of the feature vectors, the best solution was obtained in the experiment combining PDH and LDA. The feature vector had only two elements and the percentage classification accuracy reached 95.2% for multi-rooted teeth class, and 92.5% for single-rooted

Shape Descriptor	Class	Acc.	PCA	
			input	output
2DFD	Multi-rooted	60.5%	15×15 subpart	2
	Single-rooted	94.2%	5×5 subpart	5
CSS	Multi-rooted	93.4%	10×10 subpart	6
	Single-rooted	95.7%	75×75 subpart	10
GFD	Multi-rooted	69.7%	5×5 subpart	2
	Single-rooted	93.7%	25×25 subpart	7
UNL-F	Multi-rooted	80.0%	100×100 subpart	3
	Single-rooted	94.0%	15×15 subpart	7
PDH	Multi-rooted	91.2%	75 bins	1
	Single-rooted	92.5%	200 bins	2
ZM	Multi-rooted	60.3%	9th order	1
	Single-rooted	94.5%	8th order	4

Table 2: The experiments utilizing shape descriptors and the Principal Component Analysis.

Shape Descriptor	Class	Acc.	LDA	
			input	PCA
2DFD	Multi-rooted	61.2%	10×10 subpart	2
	Single-rooted	96.9%	5×5 subpart	6
CSS	Multi-rooted	79.8%	10×10 subpart	5
	Single-rooted	81.9%	20×20 subpart	7
GFD	Multi-rooted	78.2%	5×5 subpart	2
	Single-rooted	92.5%	5×5 subpart	1
UNL-F	Multi-rooted	77.0%	5×5 subpart	3
	Single-rooted	83.9%	15×15 subpart	7
PDH	Multi-rooted	95.3%	200 bins	3
	Single-rooted	92.5%	200 bins	2
ZM	Multi-rooted	64.0%	8th order	8
	Single-rooted	94.5%	10th order	4

Table 3: The experiments utilizing shape descriptors and the Linear Discriminant Analysis.

teeth class. Consequently, this approach is regarded as the best solution for classifying molars.

5 CONCLUSIONS

In this paper, an approach for tooth shapes classification to multi-rooted and single-rooted teeth classes is proposed. The approach utilizes a combination of six various shape descriptors and three different data reduction techniques. The experiments were performed using 903 test objects and 10 templates, where one class was represented by five templates. Test objects were extracted from 30 panoramic radiographs and templates were extracted from other randomly selected orthopantomograms. In order to assign a class to a test object, the Euclidean distance between a particular test object's description vector and all templates' description vectors was calculated. The nearest template indicated a class of the test object. The experimental results were evaluated in terms of two factors: the highest classification accuracy and the smallest description vector size. The best results were obtained for PDH+LDA with the accuracy of 95.2% for multi-rooted teeth classification and 92.5% for single-rooted teeth classification. The results are satisfactory, however further improvements are still necessary.

It is important to emphasize the purpose of such classification. Panoramic radiographs are not as popular in human identification as bitewing radiographs, probably due to teeth occlusion and blurry areas. However, they still form a good source of dental data, and in some cases may be the only source available. The proposed classification approach divides teeth into two groups. Knowing that molars have more diversified shapes, the binary tooth images classified as multi-rooted teeth can be applied as a database for person identification. In this case, the proposed approach plays a role of a coarse classification and a search space reduction, which are performed before an exact identification.

	Multi-rooted class		
	original	PCA	LDA
2DFD variant	61.5% 2 × 2	60.5% 2	61.2% 2
CSS variant	81.4% 10 × 10	94.0% 6	79.8% 2
GFD variant	61.8% 2 × 2	69.0% 2	78.2% 2
UNL-F variant	68.8% 5 × 5	80.0% 6	77.0% 2
PDH variant	72.0% 2 bins	91.2% 1	95.5% 2
ZM variant	60.2% 1st order	60.3% 1	64.0% 2

Table 4: Summary results for multi-rooted class.

	Single-rooted class		
	original	PCA	LDA
2DFD variant	93.7% 15 × 15	94.2% 5	96.6% 2
CSS variant	73.5% 25 × 25	95.7% 9	81.9% 2
GFD variant	93.2% 10 × 10	93.7% 7	92.5% 2
UNL-F variant	86.2% 15 × 15	94.0% 9	83.9% 2
PDH variant	76.0% 50 bins	92.4% 2	92.5% 2
ZM variant	94.0% 8th order	94.5% 4	94.5% 2

Table 5: Summary results for single-rooted class.

6 REFERENCES

[Abb99] Abbasi, S., Mokhtarian, F., Kittler, J. Curvature scale space image in shape similarity retrieval. Multimedia Systems, Vol. 7, pp. 467–476, 1999.

[Als12] Al-sherif, N., Guodong Guo, Ammar, H.H. Automatic classification of teeth in bitewing dental images using OLPP. 2012 IEEE International Symposium on Multimedia (ISM), pp. 92–95, 2012.

[Ari12] Arifin, A.Z., Hadi, M., Yuniarti, A., Khotimah, W., Yudhi, A., Astuti, E.R. Classification and numbering on posterior dental radiography using support vector machine with mesiodistal neck detection. 2012 Joint 6th International Conference on Soft Computing and Intelligent Systems (SCIS) and 13th International Symposium on Advanced Intelligent Systems (ISIS), pp. 432–435, 2012.

[Bar12] Barboza, E.B., Marana, A.N., Oliveira, D.T. Semiautomatic Dental Recognition Using a Graph-Based Segmentation Algorithm and Teeth Shapes Features. Proceedings of 5th IAPR International Conference on Biometrics (ICB), pp. 348–353, 2012.

[Che08] Chen, H., Jain, A.K. Automatic Forensic Dental Identification, in: Jain, A.K., Flynn, P., Ross, A.A. (eds.) Handbook of Biometrics, pp. 231–251, 2008.

[Che09] Chen, H., Jain, A.K. Dental Biometrics, in: Li, S.Z., Jain, A.K. (eds.), Encyclopedia of Biometrics, Springer US, pp. 216–223, 2009.

[Cun07] Cunningham, P. Dimension Reduction, Technical Report UCD-CSI-2007-7, 2007.

[Fod02] Fodor, I.K. A survey of Dimension Reduction Techniques. U.S. Department of Energy, Lawrence Livermore National Laboratory, 2002.

[Fre10a] Frejlichowski, D. An Experimental Comparison of Three Polar Shape Descriptors in the General Shape Analysis Problem, in: Świątek, J., Borzemski, L., Grzech, A., Wilimowska, Z. (Eds.), Information Systems Architecture and Technology—System Analysis Approach to the Design, Control and Decision Support, pp. 139–150, 2010.

[Fre10b] Frejlichowski, D., Wanat, R. Application of the Laplacian Pyramid Decomposition to the Enhancement of Digital Dental Radiographic Images for the Automatic Person Identification, in: Campilho, A., Kamel, M. (Eds.), ICIAR 2010, Part II, LNCS 6112, pp. 151–160, 2010.

[Fre11a] Frejlichowski, D., Wanat, R. Automatic Segmentation of Digital Orthopantomograms for Forensic Human Identification, in: Maino, G., Foresti, G.L. (Eds.), ICIAP 2011, Part II, LNCS 6979, pp. 294–302, 2012.

[Fre11b] Frejlichowski, D., Wanat, R. Extraction of Teeth Shapes from Orthopantomograms for Forensic Human Identification, in: Berciano, A. et al. (Eds.), CAIP 2011, LNCS 6855, pp. 65–72, 2011.

[Gho12] Ghodsi, S.B., Faez, K. A Novel Approach for Matching of Dental Radiograph Image Using Zernike Moment. Proceedings of 2012 IEEE International Conference on Computer Science and Automation Engineering 3, pp. 303–306, 2012.

[Gra00] Gray, H. Anatomy of the human body, 20th ed., Philadelphia: Lea & Febiger, 1918; Bartleby.com, 2000 [online] http://www.bartleby.com/107/

[Jai05] Jain, A.K., Chen, H. Registration of Dental Atlas to Radiographs for Human Identification. Proceedings of SPIE Conference on Biometric Technology for Human Identification II 5779, pp. 292–298, 2005.

[Kuk98] Kukharev, G. Digital Image Processing and Analysis (in Polish), SUT Press, Stettin, 1998.

[Kuo10] Kuo, C.H., Lin, P.L. An effective dental work extraction and matching method for bitewing radiographs. 2010 International Computer Symposium (ICS), pp. 495–499, 2010.

[Mah05] Mahoor, M.H., Abdel-Mottaleb, M. Classification and numbering of teeth in dental bitewing images. Pattern Recognition, Vol. 38, pp. 577–586, 2005.

[Mar11] Marana, A.N., Barboza, E.B., Papa, J.P., Hofer, M., Oliveira, D.T. Dental Biometrics for Human Identification, in: Midori, A. (ed.), Biometrics—Unique and Diverse Applications in Nature, Science, and Technology, InTech, pp. 41–56, 2011.

[Nas08] Nassar, D.E., Abaza, A., Li, X., Ammar, H. Automatic Construction of Dental Charts for Postmortem Identification. IEEE Transactions on Information Forensics and Security, Vol. 3, No. 2, pp. 234–246, 2008.

[Pat12] Pattanachai, N., Covavisaruch, N., Sinthanayothin, C. Tooth recognition in dental radiographs via Hu's moment invariants. 9th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, pp. 1–4, 2012.

[Phi09] Phillips, V.M., Stuhlinger, M. The Discrimination Potential of Amalgam Restorations for Identification: Part 1. The Journal of Forensic Odonto-stomatology, Vol. 27, pp. 17–22, 2009.

[Pre01] Pretty, A., Sweet, D. A Look at Forensic Dentistry—Part I: The Role of Teeth in the Determination of Human Identity. British Dental Journal, Vol. 190, pp. 359–366, 2001.

[Raj11] Raju, J., Modi, C.K. A proposed Feature Extraction Technique for Dental X-Ray Images Based on Multiple Features. 2011 International Conference on Communication Systems and Network Technologies, pp. 545–549, 2011.

[Rau94] Rauber, T.W. Two Dimensional shape description, Technical report: GR UNINOVA-RT-10-94. Universidade Nova de Lisboa, Lisoba, Portugal, 1994.

[Sul15] Sultana, M., Paul, P.P., Gavrilova, M. Occlusion Detection and Index-based Ear Recognition. Journal of WSCG, Vol. 23, No. 2, pp. 121-129, 2015.

[Vin08] Viner, M.D. The use of radiology in mass fatality events, in: Adams, B., Byrd, J. (eds.) Recovery, Analysis, and Identification of Commingled Human Remains, Humana Press, Totowa, New Jersey, pp. 145–184, 2008.

[Yan08] Yang, M., Kpalma, K., Ronsin, J. A survey of shape feature extraction techniques. In: Yin, P.Y. (ed.) Pattern Recognition Techniques, Technology and Applications, InTech, pp. 43–90, 2008.

[Yan10] Yan, H.B., Jin, A.T.B., Yin, O.S., Aziz, F.F.A. A secure touch-less based fingerprint verification system. Journal of WSCG, Vol. 18, No. 1-3, pp. 1–8, 2010.

[Yun12] Yuniarti, A., Nugroho, A.S., Amaliah, B., Arifin, A.Z. Classification and Numbering of Dental Radiographs for an Automated Human Identification System. TELKOMNIKA, Vol. 10, No. 1, pp. 137–146, 2012.

[Zha02] Zhang, D., Lu, G. Shape-Based Image Retrieval Using Generic Fourier Descriptor. Signal Processing: Image Communication, Vol. 17, No. 10, pp. 825–848, 2002.

Inter-Pixel Filtering of Digital Images With OpenCL on NVIDIA GPUs

José Valero
Francisco José de Caldas District University
Spatial Data Research Group (NIDE)
Engineering Faculty
110231, Bogotá, Colombia
jvalero@udistrital.edu.co

ABSTRACT

The process of filtering digital images represented by complex Cartesian allows to use the available one-dimensional (1D) elements (interpixel); however, having those additional 1D elements increases both the volume of data and the time for processing them. The time reduction strategy based on a parallel computing scheme on the number of available central processing units (CPUs) does not consider additional computing resources such as those offered by general purpose graphics processing units (GPUs) of NVIDIA. Parallel computing possibilities provided by the NVIDIA GPUs were explored and, based on them, a computational scheme for the digital image Cartesian complexes filtering task was proposed using the application program interface Open Computing Language (OpenCL) provided for NVIDIA corporation GPUs. The results assessment was established by comparing the response times of the proposed solution compared to those obtained using only CPU resources. The obtained implementation is an alternative to parallelization of the filtering task, which provides response times up to 14 times faster than those obtained with the implementation that uses only the CPU resource. The NVIDIA multicore GPU significantly improves the parallelism, which can be used in conjunction with the available multicore CPU computing capacity, balancing the workload between these two computing powers using both simultaneously.

Keywords

Cartesian complex, digital image, filtering, NVIDIA, OpenCL.

1 INTRODUCTION

The increasing availability of high geometric, radiometric, spectral and temporal resolution multispectral digital images of the earth surface increases the possibility of using them in space planning and management applications geographical. Such applications require object detection capabilities for monitoring based on edge detection procedures very frequently supported in the outcomes of the application of filters for edge detection, which, given the high volumes of data to be processed, demand huge computing resources. So, computational implementations, which are focused on reducing response times, are required.

In recent decades, specific purpose parallel computing capabilities for graphics processing have evolved, giving rise to graphical processing units (GPUs), which can be leveraged for other processing, when conveniently using application programming interfaces (APIs) they provide [1]. Efforts have been made to provide strategies and environments that facilitate programming to enhance GPU-based software development.

In [2] an introduction to modern PC architectures is given and discusses strategies and guidelines for developing GPU programs. [3] introduces a graphical user interface (GUI) tool called GPUBlocks whose purpose is to facilitate parallel programming in multicore computer systems. The GPU is designed for high-speed graphic processing that is inherently parallel. The Open Computing Language (OpenCL) takes a platform independent simple model of data parallelism and incorporates it into a programming model [4]. The OpenCL language makes the GPU look the same as another programmable device, using an execution model defined in terms of device kernels and a host program. In this way, the executable is in a virtual platform independent architecture, The OpenCL framework. This framework allows applications to use a host and one or more OpenCL devices as a single heterogeneous parallel computer system. Backwards compatibility allows

https://doi.org/10.24132/CSRN.2020.3001.3

that a device consume earlier versions of the OpenCL C and other programming languages [5].

[6] manages to design and implement an alternative method for segmenting multispectral images based on axiomatic locally finite spaces (ALFS) provided by Cartesian complexes, and which takes into account topological and geometric properties [7]. This alternative representation model provides a geometric space that complies with the digital topology T_0 free of topological ambiguities, on which a new way of segmenting image data is built. The proposed model is developed and implemented in such a way that required topological and geometric characteristics present in the combinatorial semi-spaces are transformed into combinatorial structures which encode them using their associated oriented matroid [8, 9, 10]. Having additional inter-pixel elements means that the volume of data increases considerably. In particular, the task of texture analysis demands a significant amount of computing resources. The strategy adopted in [6], based on a parallel computing scheme from the execution in as many groups of 1-cells as the number of central processing units (CPUs) available, significantly reduces the time of processing required to process the entire Cartesian complex. However, the size of the digital image makes up such amount of data to be processed that the response times, even with dozens of CPUs, are prolonged.

[11] examines parallel algorithms for performing image boundary detection as part of the segmentation process and performs deployment on highly parallel NVIDIA processors, achieving a contour detector that provides accuracy with an F-measure of 0.70. The runtime is reduced from 4 minutes to 1.8 seconds. The efficiency gains that are made there allow the detection of contours in much larger images and the algorithms are applicable to various approaches to image segmentation. [12] starts from the fact that nowadays it is increasingly common to detect changes in land use and coverage using multispectral images and that a large part of the available change detection (CD) methods available focus on pixel-based operations. Since the use of spectral-spatial techniques helps to improve accuracy results, but also implies a significant increase in process time, [12] used a GPU framework to make object-based CD from multitemporal hyperspectral images and achieved real-time execution with accelerations of up to 46.5 times with respect to an open multiprocessing (OpenMP) implementation.

This research proposes the implementation of the multi-kernel multi-scale steerable filtering task of digital images represented as Cartesian complexes, on graphical processing units (GPUs), using a parallel computational scheme in terms of the OpenCL API. In this way, the processing that is currently assigned to each CPU can be performed in parallel including also the processing cores available on the GPU. The evaluation of the results was established by comparing the response times of the proposed solution compared to those obtained using only CPU resources. The produced implementation is an alternative parallelization of the filtering task, which provides response times up to 26 times faster than those obtained with the proposed implementation in [6].

2 PARALLELIZATION SCHEMA

The possibilities of parallel computation provided by the GPUs were explored in the context of the multi-kernel multi-scale steerable filtering task proposed in [13] and reformulated in [6] in terms of Cartesian complex. Finally, a parallel computational scheme was proposed and the outcomes quality assessment was established by comparing the response times of the proposed solution against the response times obtained through the implementation proposed in [6].

2.1 RAM memory resource management scheme

Memory in OpenCL is divided into Host and Device Memories. The device memory is directly available to kernels executing on OpenCL devices. In fact, a OpenCL kernel through work-items can access data from private memory spaces, local memory that can be used to allocate variables that are shared by all work-items in a work-group. Additionally, all work-items in all work-groups running on any device within a context can access to any element of a memory object in both Global and Constant Memories [5]. Therefore, it is unavoidable to transfer data from the host memory space to the device memory for processing and from the device memory to the host memory to obtain the results.

In order to reduce the copying time, the OpenCL function call for non-blocking memory transfers between host-device are used, by submitting Memory commands to a command-queue. So, the waiting times of the data copying between device and host memories in non-blocking mode can concurrently be used to perform other tasks on the host side. Based on the above, the required global device memory must be fully allocated to copy to there the input data from the host memory and then passed to each device work-item the segments that it must copy to the shared local memory, process and , after getting the results, copy from the shared local memory to the global device memory. There are two sets of data that must be copied in this way: (i) the one-dimensional cells of the Cartesian complex space over which the filtering process is to be performed; and (ii) the set of multi-kernel and multi-scale steerable filters [6] that will be applied. The two data sets will remain in the device memory shared by the threads within a block, which involves the intra-device copying between global and shared memories.

If GPU device provides support for concurrent copying, the transfer time between memory spaces becomes a load balancing parameter, being clear that whenever possible it should be performed simultaneously with data subset processing assigned to the threads available on the CPU. On the other hand, the use of the parallel computation processing available in the GPU can be incorporated into the processing of the functions of the multi-kernel and multi-scale oriented filtering task of Cartesian complexes based on at least two parallelization schemes: (i) considering each CORE of the GPU as an additional processing unit and extending to them the parallelism scheme proposed in [6], and (ii) creating as many GPU work-items as one-dimensional space cells present in the Cartesian complex corresponding to the image to be processed and leave parallelization management to the GPU. Regardless of the schema used, the code block that the kernel must execute must present as few branches as possible in order to increase the performance of the underlying work-items within a single work-group execute concurrently guaranteed to make independent progress in the presence of subgroups and device support [5]. Next, the algorithmic solution for each of the two parallelization schemes mentioned above will be presented.

2.2 Parallelism scheme considering each GPU core as a processing unit equivalent to a CPU core

One intended parallelism scheme using the highly massive capabilities provided by a GPU is to treat each GPU core as an equivalent CPU core. In parallelism scheme proposed in [6], the one-dimensional cells of the Cartesian complex space are distributed using the row prime path [14, pp. 233, 234], among the m available CPU cores (see Fig.1 left). What is proposed here is to make the distribution of the one-dimensional cells taking each one of the p GPU cores that are available, as an additional processing unit, for a total of $m+p$ processing units (see Fig.1 right). However, in the case of the GPU, the following should be taken into account: first, the necessary copies between the host and device memory resources must be made; and second, differentiated code bifurcations between work-items of the same work-group should be avoided [15].

In this parallelism scheme, there is a one-dimensional NDRange consisting of as many work-items as the number of cores available on the device (GPU). In order to avoid code divergence between work-items that are part of each work-group eventually released by the GPU, the number of one-dimensional cells assigned to each GPU core must be the same, so the expression for the size calculation of each group of one-dimensional cells proposed in [6, p. 98] should be rethought.

A balanced mapping is made between the CPU and GPU, with an initial distribution based on the proportion that the number of GPU cores represents of the total available processing units. So, the number pct_{GPU} of one-dimensional cells proportionally assigned to the GPU is set based on equation Equation (1), and number g_1 of one-dimensional cells assigned to each core based on Equation (2).

$$pct_{GPU} = ceil\left(r\frac{p}{m+p}\right) \times c \,, \tag{1}$$

where r and c are respectively the number of rows and columns in the image, while m and p are respectively the number of CPU cores and GPU cores. $ceil$ means "the smallest integer larger than".

$$g_1 = \frac{pct_{GPU} - \left(pct_{GPU}\%p\right)}{p} \,, \tag{2}$$

where % represents the module operation. The number of cells assigned to the GPU n_{GPU} that guarantees the same number of cells for each of its threads is given by Equation (3).

$$n_{GPU} = g_1 \times p \,. \tag{3}$$

So, the number of one-dimensional cells assigned to the CPU n_{CPU} is set according to Equation (4).

$$n_{CPU} = (r \times c) - n_{GPU} \,. \tag{4}$$

Finally, the distribution scheme proposed in [6, p. 98] is applied to obtain the two group sizes applicable to the CPU cores according to Equations (5) and (6).

$$g_2 = \frac{n_{CPU} - \left(n_{CPU}\%m\right)}{m} \,. \tag{5}$$

$$g_3 = g_2 + 1 \,. \tag{6}$$

Therefore, each of the p cores of GPU will get assigned g_1 one-dimensional cells, and there will be as many as $m - (n_{CPU}\%m)$ CPU cores with g_2 assigned one-dimensional cells and as many as $(n_{CPU}\%m)$ CPU cores with g_3 assigned one-dimensional cells.

2.3 Parallelism scheme creating as many GPU threads as cells assigned to the device

The second tested parallelism scheme, using the highly massive capabilities provided by a GPU, is to create as many GPU work-items as the number of

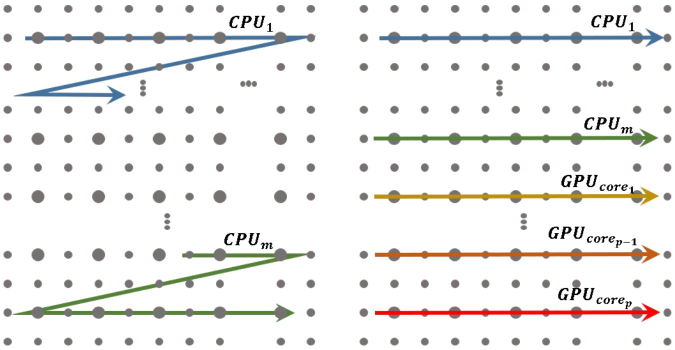


Figure 1: Only CPU versus GPU and CPU parallelism schemes.

two-dimensional cells assigned to it. This scheme uses the same group sizes used in section 2.2 by Equations (2), (5) and (6) but instead of creating p device work-items each with g_1 two-dimensional cells, $p \times g_1$ work-items are created in the device, each processing a cell of the Cartesian complex as shown in Figure 2 (bottom). As can be seen in the figure (above), the cells assigned to the CPU are distributed among the available cores using the scheme proposed in [6, p. 98].

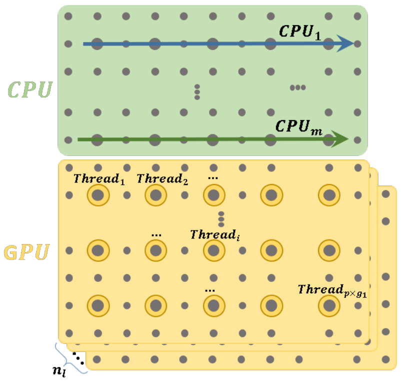


Figure 2: One GPU thread by two-dimensional cell parallelism scheme.

As in the scheme proposed in section 2.2, in the case of the GPU, the following should be considered: first, that the necessary copies between the host and device memory resources must be made; and second, that differentiated code bifurcations between work-items of the same work-group should be avoided. In this parallel scheme, a three-dimensional NDRange whose first and second dimension sizes correspond respectively with the number of rows and columns of the two-dimensional cells set assigned to the GPU, and the third dimension corresponds to n_l layers in the digital image to be processed. In this case the divergence of the code between the work-items that are part of each work-group, is counteracted by assigning to each work-item only one cell of the Cartesian complex in each layer of the digital image. Since the position of the two-dimensional cell assigned to a work-item is determined by the position of the work-item within the hierarchy of work-items, the number of cells must be a multiple of the number of columns in the image. By the above, in this case the number of two-dimensional cells assigned to the GPU is defined based on Equation 1 but using Equation 7 instead of Equation 3.

$$n_{GPU} = ceil\left(r \times c \frac{p}{m+p}\right). \qquad (7)$$

3 PARALLELISM MODEL IMPLEMENTATION

For performance reasons, the parallelization model of the multi-scale, multi-kernel steerable filtering process proposed in Section 2 using the OpenCL API, was adapted to Nvidia's CUDA programming framework and implemented using the integrated development environment (IDE) provided by Eclipse [16], for the Java and C/C++ programming languages. The Java language was used to include in the implementation proposed in [6] the copying of the host memory to that of the device (and vice versa) and for the configuration of the grid of threads blocks that are launched in the GPU using the respective support provided by the JCuda API version 10.1 [17]. The C / C ++ language was used to implement the kernel code that is executed by each thread. Next, a description of these components is made.

3.1 Host - device memory copying and thread block grid configuration

The implementation of the host - device memory copy scheme proposed in Section 2.1, included two phases: (i) the first phase includes copying the input data from the host memory to that of the device before launching the Thread block grid and the final copy of the results from the device memory to the host; and (ii) the second phase comprises the copying of the input data from the device memory to the memory shared by the threads of each block, as well as the final copying of the results from the memory shared to the device memory.

Copying between host and device memory was carried out concurrently for the four data sets: (i) the Cartesian input complex, (ii) the filter bank to be applied, (iii) the arrangement with the number of 2-cells (pixels) in each filtering kernel, and (iv) the resulting Cartesian complex. This was implemented as shown in the code fragments of the filtering process shown in the Java program listing 1.

Because the JNI API used by JCuda to pass the parameters in the components implemented in Java and C programming languages only allows the use of one-dimensional arrays, it was necessary to convert the input data sets to a representation of one-dimensional primitive data arrays and vice versa (lines 8 and 11). To carry out this copying process, the asynchronous memory copy support provided by the GPU was used. The asynchronous copying process began by creating a stream which was assigned in the creation of the page-locked memory segments of the host and device (lines 12, 14, 17, 19, 21, 24 and 31), in asynchronous copying from conventional memory to host page-locked memory (lines 13 and 15), from host memory to device memory (lines 18 and 20), as in those copied from the memory of the device to that of the host (line 32). To ensure that the asynchronous copying process was carried out consistently, before proceeding to a next phase, synchronization points were included (lines 16, 24, 30 and 33).

Copying to the shared memory segments was configured using the support provided by JCuda in launching the thread block grid (lines 26 to 29), defining the size of the memory segment shared by the threads within a block (line 28) and copying in it the input data for each GPU thread. The kernels that must be executed by each GPU thread, according to the massively concurrent processing schemes proposed in Sections 2.2 and 2.3, were implemented in C programming language in the way described below.

3.2 Kernel code implementation considering each GPU core as an additional processing unit

In the GPU Kernel listing 2, shown below, presents the GPU kernel that must be executed by each thread for the parallelism scheme defined in Section 2.2.

Each kernel starts by defining the memory segment shared by the threads that are members of a block (line 27), in which the input data that will be used in the filtering process is placed (line 30), seeking to use the memory management that guarantees the greatest efficiency. In contrast to the filtering process implemented in [6], it acts not on two-dimensional but on one-dimensional data structures, so that their indexing had to be adapted (lines 5, 6, 13, 23 and 34). On the other hand, in order to avoid the divergence of threads warps, the "if" sentences were completely replaced by logical expressions that evaluate to 1 or 0 used to adequately activate addends or factors in accumulation expressions (line 14). These logical expressions were also used to activate / deactivate displacements in indexing expressions (lines 14 and 23). In this way, the only flow control statement used is “for”, but assigning to each GPU thread the same number of one-dimensional cells, which guarantees that the number of cycles in each thread that is part of A warp is exactly the same.

3.3 Kernel code implementation for a GPU thread for each two-dimensional cell

In the GPU Kernel listing 3, shown below, presents the GPU kernel that must be executed by each thread for the parallelism scheme defined in Section 2.3.

Each kernel starts by defining the memory segment shared by the threads in a block (lines 19 and 21), in which the input data that will be used in the filtering process is placed (lines 25 and 26), looking for use memory management that guarantees the highest efficiency. As in the parallelism scheme implemented in Section 3.2, it acts not on two-dimensional data structures but on one-dimensional data, so, its indexing had to be adapted (lines 3, 4, 8, 9 and 10). On the other hand, with the purpose of avoiding the divergence of thread warps, the “if” sentences were completely replaced by logical expressions that evaluate values 1 or 0 used to adequately activate addends or factors in accumulation expressions (line 10); as well as displacements in indexing expressions (line 15). Additionally, the multidimensionality of the GPU threads hierarchy was used, using the thread X dimension (threadIdx.x) and thread blocks (blockIdx.x) to access the 2-cells of the input Cartesian complex, the Y dimension of the thread block grid (blockIdx.y) to access each filtering kernel, and the Z dimension of the thread block grid (blockIdx.z) to Access each input channel. This allowed the removal of the respective “for” cycles of the DirectionalFilterProcessor, computeFilteredCell and applyFilter functions, leaving only one “for” cycle in the applyFilter function to control the filtering kernel neighborhood.

Thus, unlike the parallelism scheme implemented in Section3.2, the only flow control statement used is a “for” cycle which is evidenced in the algorithmic simplicity of the implementation presented in the GPU Kernel listing 3 regarding the one presented in the listing 2. The “if” flow control statement used on line 39 does not introduce significant divergence of threads as it is used exclusively to avoid the process of threads that are not assigned 2-cells.

4 RESULTS AND DISCUSSION

The implementation described in Section 3, was executed on a machine with OpenSUSE linux Leap 15.1

Listing 1: Java copying code snippet for FilteredCartesianComplex method

```java
public FilteredCartesianComplex(...) {
  int cores = Runtime.getRuntime().availableProcessors();
  // Enable exceptions and omit all subsequent error checks
  JCudaDriver.setExceptionsEnabled(true);
  String ptxFileName = preparePtxFile("DirectionalFilterProcessorByCore.cu");
  cuInit(0); CUdevice device = new CUdevice();
  ...
  inputImage = aCartesianComplex.convert2PrimitiveArray(rows4GPU);
  hostInputImage = new Memory(MemoryType.PAGE_LOCKED,...,stream);
  hostInputImage.put(inputImage);
  float inputFilterBank[] = convert2PrimitiveArray(nFilterKernel, ...);
  Memory hostInputFilterBank = new Memory(MemoryType.PAGE_LOCKED, ...);
  hostInputFilterBank.put(inputFilterBank);
  Memory hostFilterKernelsCellCount = new Memory(MemoryType.PAGE_LOCKED,...);
  hostFilterKernelsCellCount.put(filterKernelsCellCount);
  cuStreamSynchronize(stream);
  deviceInputImage = new Memory(MemoryType.DEVICE, sizeOf2CellValues,stream);
  cuMemcpyHtoDAsync(deviceInputImage , hostInputImage, ...);
  deviceFilterBank = new Memory(MemoryType.DEVICE, ..., stream);
  cuMemcpyHtoDAsync(deviceFilterBank, ..., stream);
  devFilterKernelsCellCount = new Memory(MemoryType.DEVICE, ..., stream);
  cuMemcpyHtoDAsync(devFilterKernelsCellCount, ..., stream);
  deviceOutputImage = new Memory(MemoryType.DEVICE, ..., stream);
  cuStreamSynchronize(stream);
  ...
  cuLaunchKernel(function,1/gridSizeX/, 1, 1, // Grid dimension
   blockSizeX , 1, 1, // Block dimension
   inputFilterBank.length * Sizeof.FLOAT, null, // Shared memory size and stream
   kernelParameters, null); // Kernel- and extra parameters
  cuCtxSynchronize();
  Memory hostOutputImage = new Memory(MemoryType.PAGE_LOCKED, ..., stream);
  cuMemcpyDtoHAsync(hostOutputImage, deviceOutputImage, ..., stream);
  cuStreamSynchronize(stream);
        hostOutputImage.get(outputImage);
  cuStreamSynchronize(stream);
 ...
}
```

operating system, with *AMD FX*™ – 8320 processor with 8 cores, RAM memory (host memory) of 16 *GiB* and a GPU device with the configuration features shown in Table 1. With the purpose of dedicating the GPU to the filtering process only, the machine was equipped with a second ATI Radeon 3000 video card for the management of the Graphic interface.

For the multi-scale multi-kernel steerable filtering process it was used a 500 by 500 pixel image. The implementation execution was monitored using NVIDIA Nsight Systems software version 2019.5.2, taking the events of creation and destruction of the GPU context, respectively as the start and end of the filtering process. The monitoring the execution of the filtering process using only the 8 CPU cores showed that (Figure is not shown for reasons of space), the filtering process starts at 1.48594 seconds, after the end of the CUDA context creation event. The filtering process ends at 2.18768 seconds, before the start of the CUDA context destruc-

Nvidia Device	GeForce GTX 950
CUDA Driver Version	10.1
CUDA Capability	5.2
Total global memory	2002 MBytes
Multiprocessors (MP)	6
CUDA Cores / MP	128 (768)
Total shared memory	49152 bytes (per block)
Warp size	32
Max block size(x, y, z)	(1024, 1024, 64)
Max grid size (x, y, z)	(2147483647,65535,65535)
Concurrent copy and kernel execution	Yes, 2 copy engine (s)

Table 1: GPU hardware configuration of the test environment.

Listing 2: process that employs one GPU core per 2-cell group

```
extern "C" _device_ void applyFilter(int central1DCoord, ...) {
 int aCoord1D = 0;

 for (int channel = 0 ; channel < numberOfInputChannels ; channel++) {
  int inputChannelCentral1DCoord = channel * numberOfInputCellTypes * pixelsInX *
    pixelsInY + central1DCoord;
  int outputChannelCentral1DCoord = ((kernelIndex * numberOfInputChannels +
    channel) *
  numberOfOutputCellTypes + cellType) * pixelsInX * pixelsInY + central1DCoord;
  aFilteredCC[outputChannelCentral1DCoord] = 0;

  for (int cellInd = 0 ; cellInd < kernelCellCount ; cellInd++) {
   int cX = (aFilterKernel[cellInd * 3] - 1 + cellType) / 2,
    cY = (aFilterKernel[cellInd * 3 + 1] - cellType) / 2;
   aCoord1D = central1DCoord + (cY * pixelsInX) + cX;
   aFilteredCC[outputChannelCentral1DCoord] += ((1 && (aCoord1D >= 0) && (aCoord1D
    < (pixelsInY * pixelsInX))) * (aMonochromatic[inputChannelCentral1DCoord + (1
    && (aCoord1D >= 0) && (aCoord1D < (pixelsInY * pixelsInX))) * ((cY * pixelsInX)
    + cX)] * aFilterKernel[cellInd * 3 + 2]));}
 }
}
extern "C" _device_ void computeFilteredCell(int  central1DCoord, ...) {
 float * dF = nFilterKernel + cellType * filterKernelsCellCount[0] * 3;

 for(int iFK= cellType ; iFK < numberOfKernels ; iFK += 2){
  applyFilter(central1DCoord, ...);
  dF = dF + (filterKernelsCellCount[iFK] + filterKernelsCellCount[(iFK + 1) %
    numberOfKernels]) * 3;}
}
extern "C" _global_ void DirectionalFilterProcessor(...){
 extern _shared_ float interchangedData[];
 float * sharedFilterBank = interchangedData;
 int groupStart =  threadIdx.x * groupSize;
 if (threadIdx.x == 0) memcpy(sharedFilterBank, nFilterKernel, sizeof(float) *
    bankSize);
 __syncthreads();

 for(int i = 0; i < groupSize; i++){
  int central1DCoord = groupStart + i;
  computeFilteredCell(central1DCoord, HORIZONTAL_CRACK, ...);
  computeFilteredCell(central1DCoord, VERTICAL_CRACK, ...);}
}
```

tion event. That is, the filtering process performed concurrently only by the eight CPU cores, each processing 31250 or 31251 pixels according to Equations 5 and 6, took a time of 701.74 milliseconds.

On the other hand, Figure 3 presents the results of the monitoring of the filtering process execution using, in addition to the 8 CPU cores, also the 768 GPU cores. In this case, of the $500 \times 500 = 250,000$ pixels, $247,296$ were assigned to the GPU, that is, 768 groups (one for each GPU core), each of 322 pixels. As shown in the figure, the filtering process starts at 1.49453 seconds, after the end of the CUDA context creation event. As can also be seen, the filtering process ends at 1.64496 seconds, before the start of the CUDA context destruction event (labeled in the figure as call to cuCtxDestroy). Therefore, the filtering process performed concurrently by the eight CPU cores, each processing 338 or 339 pixels according to equations Equations 5 and 6, took a time of 150.43 milliseconds. In the case of GPU threads, the filtering process performed concurrently by processing each 322 pixels, started at 1.52946 seconds and finished 1.58076 (labeled in the figure as DirectionalFilterProcessor), that is, it took 51.3 milliseconds. This means that, using all available CPU and GPU computing resources, the process performed by the CPU

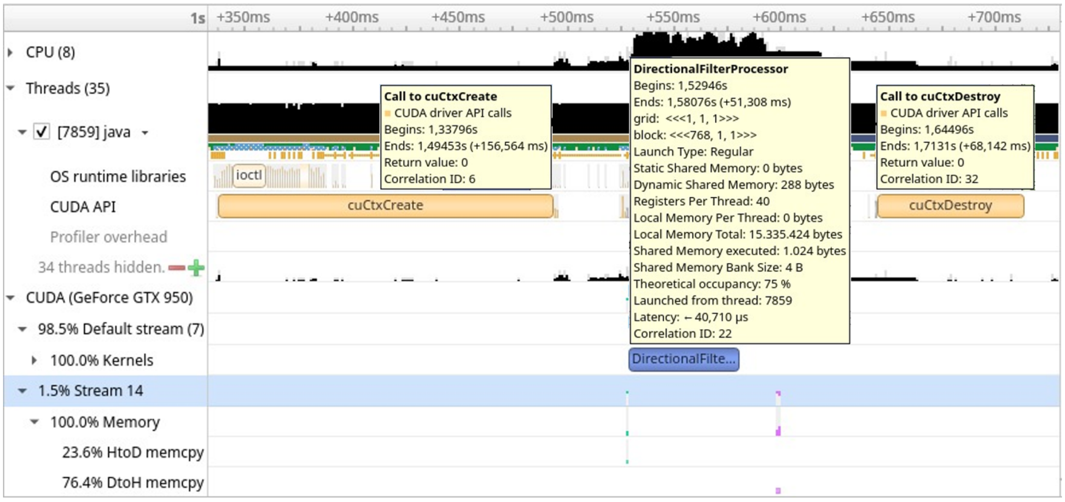
Listing 3: process that employs one GPU thread per 2-cell

```
extern "C" _device_ void applyFilter(int central1DCoord, ...) {
  int aCoord1D = 0;
  int inputChannelCentral1DCoord = channel * numberOfInputCellTypes * pixelsInX *
    pixelsInY + central1DCoord;
  int outputChannelCentral1DCoord = ((kernelIndex * numberOfInputChannels +
    channel) * numberOfOutputCellTypes + cellType) * pixelsInX * pixelsInY +
    central1DCoord;
  aFilteredCC[outputChannelCentral1DCoord] = 0;

  for (int cellInd = 0; cellInd < kernelCellCount; cellInd++) {
    int cX = (aFilterKernel[cellInd * 3] - 1 + cellType) / 2, cY =
    (aFilterKernel[cellInd * 3 + 1] - cellType) / 2;
    aCoord1D = central1DCoord + (cY * pixelsInX) + cX;
    aFilteredCC[outputChannelCentral1DCoord] += ((1 && (aCoord1D >= 0) && (aCoord1D
    < (pixelsInY * pixelsInX))) *  (aMonochromatic[inputChannelCentral1DCoord + (1
    && (aCoord1D >= 0) && (aCoord1D < (pixelsInY * pixelsInX))) * ((cY * pixelsInX)
    + cX)] * aFilterKernel[cellInd * 3 + 2]));}
}
extern "C" _device_ void computeFilteredCell(int  central1DCoord, ...) {
  applyFilter(central1DCoord, ...,
    nFilterKernel + filterKernelsCellCount[2 * (2 * iFK + cellType) + 1],
    filterKernelsCellCount[2 * (2 * iFK + cellType)], pixelsInX, pixelsInY,
    channel);
}
extern "C" _global_ void DirectionalFilterProcessor(...){
  extern _shared_ float interchangedData[];
  float * sharedFilterBank = interchangedData;
  extern _shared_ float sharedFilterKernelsCellCount[numberOfKernels * 4];
  int central1DCoord = blockIdx.x * blockDim.x + threadIdx.x;

  if (threadIdx.x == 0){
    memcpy(sharedFilterBank, nFilterKernel, sizeof(float) * bankSize);
    memcpy(sharedFilterKernelsCellCount, devFilterKernelsCellCount, sizeof(float) *
    numberOfKernels * 4);}

  __syncthreads();

  if (central1DCoord < (pixelsInX * pixelsInY)){
   computeFilteredCell(central1DCoord,HORIZONTAL_CRACK, blockIdx.z,blockIdx.y,...);
   computeFilteredCell(central1DCoord,VERTICAL_CRACK, blockIdx.z,blockIdx.y,...);}
}
```

cores was $\frac{701.74}{150.43} \cong 5$ times faster, while the process performed by the GPU cores was $\frac{701.74}{51.3} \cong 14$ times faster.

On the other hand, Figure 4 presents the results of the monitoring of the filtering process execution using all computing resources, the 8 CPU cores and the 768 GPU cores. In this case, the GPU was also assigned 247,296 2-cells (pixels), however, a GPU thread was created for every 2-cell of each channel in the Cartesian input complex and for each filtering kernel. This is evidenced in Figure 4 in which can be seen the hierarchy of threads composed of a grid of threads blocks of $242 \times 2 \times 1$ and each grid of threads within each block of $1024 \times 1 \times 1$. As shown in the figure, the filtering process starts at 1.37085 seconds and ends at 1.52964. Therefore, the filtering process performed concurrently by the eight CPU cores, each processing 338 or 339 pixels, took a time of 158.79 milliseconds. In the case of GPU threads, the filtering process carried out concurrently by processing each one an input channel and a kernel of the two 1-cells of the pixel assigned to it, started at 1.40355 seconds and ended 1.43087, that is, it took 27.32 milliseconds.

This means that, using all available CPU and GPU computing resources, the process performed by the CPU cores was $\frac{701.74}{158.79} \cong 4$ times faster, while the one performed by GPU cores were $\frac{701.74}{27.32} \cong 26$ times faster.

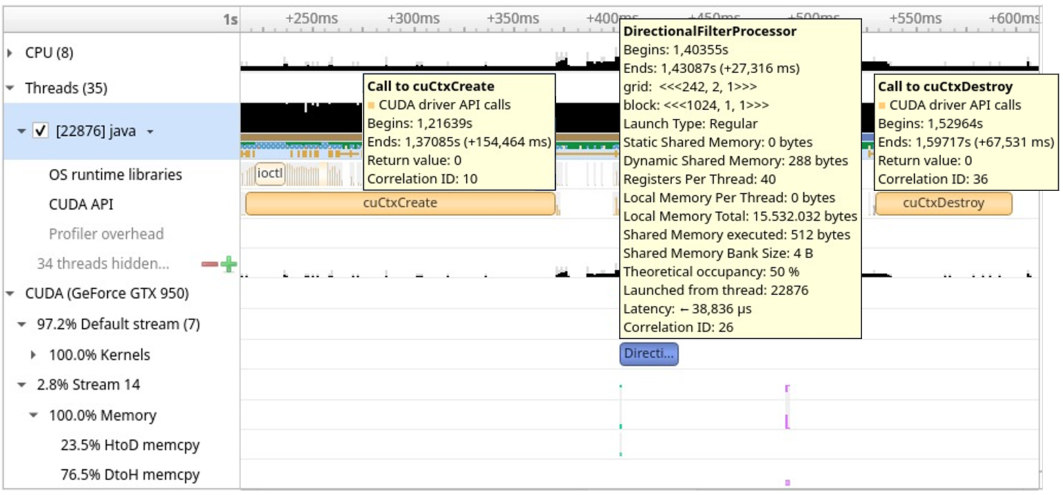


Call to cuCtxCreate
CUDA driver API calls
Begins: 1,33796s
Ends: 1,49453s (+156,564 ms)
Return value: 0
Correlation ID: 6

DirectionalFilterProcessor
Begins: 1,52946s
Ends: 1,58076s (+51,308 ms)
grid: <<<1, 1, 1>>>
block: <<<768, 1, 1>>>
Launch Type: Regular
Static Shared Memory: 0 bytes
Dynamic Shared Memory: 288 bytes
Registers Per Thread: 40
Local Memory Per Thread: 0 bytes
Local Memory Total: 15.335.424 bytes
Shared Memory executed: 1.024 bytes
Shared Memory Bank Size: 4 B
Theoretical occupancy: 75 %
Launched from thread: 7859
Latency: ←40,710 μs
Correlation ID: 22

Call to cuCtxDestroy
CUDA driver API calls
Begins: 1,64496s
Ends: 1,7131s (+68,142 ms)
Return value: 0
Correlation ID: 32

Figure 3: Monitoring of the filtering process using 8 CPU and 768 GPU cores.



Call to cuCtxCreate
CUDA driver API calls
Begins: 1,21639s
Ends: 1,37085s (+154,464 ms)
Return value: 0
Correlation ID: 10

DirectionalFilterProcessor
Begins: 1,40355s
Ends: 1,43087s (+27,316 ms)
grid: <<<242, 2, 1>>>
block: <<<1024, 1, 1>>>
Launch Type: Regular
Static Shared Memory: 0 bytes
Dynamic Shared Memory: 288 bytes
Registers Per Thread: 40
Local Memory Per Thread: 0 bytes
Local Memory Total: 15.532.032 bytes
Shared Memory executed: 512 bytes
Shared Memory Bank Size: 4 B
Theoretical occupancy: 50 %
Launched from thread: 22876
Latency: ←38,836 μs
Correlation ID: 26

Call to cuCtxDestroy
CUDA driver API calls
Begins: 1,52964s
Ends: 1,59717s (+67,531 ms)
Return value: 0
Correlation ID: 36

Figure 4: Monitoring of the filtering process using a GPU thread for every 2-cell.

The adopted hierarchy of memory scheme, which is based on the use of the device's global memory to copy there both the input image represented by the Cartesian complex, and the filter bank to be applied to each 1-Cell and then copy it to the memory shared by each block of threads, involves three phases of memory transfer, including the final copying of the results. The processing time is significantly reduced by performing it in parallel using the asynchronous copy scheme provided by the CUDA API, as well as assigning to each of the threads blocks the copying in the shared memory segments, the data to be processed. This means that, although it is true, additional time is devoted to copy processing, the overall performance is also increased when processing on the cells of the Cartesian complex, since they are placed in shared memory of the device, which has the largest bandwidth.

The massive parallelization schemes presented in Section 2, which were introduced using the Java programming language and the CUDA implementation for Java called Jcuda [17], as well as the C programming language for the case of the processing kernels to be executed by a GPU thread, show at least two ways of performing the multi-scale and multi-kernel filtering process of an image represented as a Cartesian complex, by distributing the cell processing among GPU and CPU resources highly massive available.

5 CONCLUSIONS AND FUTURE WORK

The proposed scheme for multi-scale and multi-kernel oriented filtering of an image represented as a Cartesian complex implies additional processing times dedicated to memory copying and conversion of the Cartesian complex to one-dimensional primitive arrays. However, since these processes are carried out in parallel, their impact on overall performance is negligible compared to the performance gain when performing the filtering process using the highly massive processing offered by GPU devices.

While it is true, the parallelism capability offered by a multi-core CPU is greatly exceeded by that offered by a multi-core GPU, the proposed parallelism approach manages to perform a proportional balance between these two parallel computing capabilities using both in a concurrent way. Although in general the frequency of a CPU may be slightly higher than that of a GPU, the data orientation of the latter makes it more appropriate for the processing of high volumes of data such as digital images represented as Cartesian complexes.

The proposed approach constitutes a prototype and therefore presents several limitations that must be taken into account and that would warrant the continuation of this research. On the one hand, the proposed processing contemplates the copying of the input Cartesian complex, represented as a one-dimensional primitive arrangement, to the global and shared memories of the device. However, these resources are limited, so that the size of the image that can be processed is quite restricted for practical purposes. In order to be applicable to images regardless of their size, a research should be undertaken that extends the copying component to and from the global and shared memories of the device, by partitioning the image into fragments that can be housed in these memory resources.

The approach proposed in [6, pp. 94-97] supports the types of data that the input image presents through the use of the GDAL layer ([18]); however, the approach proposed here only supports the float data type, so a research should be undertaken that addresses the types supported by GDAL layer.

6 ACKNOWLEDGMENTS

Our thanks to the District University for allowing work time to carry out the research.

7 REFERENCES

[1] T. Soyata, *GPU Parallel Program Development Using CUDA*. CRC Press, Inc., 01 2018.

[2] A. R. Brodtkorb, T. R. Hagen, C. Schulz, and G. Hasle, “Gpu computing in discrete optimization. part i: Introduction to the gpu,” *EURO Journal on Transportation and Logistics*, vol. 2, p. 129, May 2013.

[3] Y.-S. Hwang, H.-H. Lin, S.-H. Pai, and C.-H. Tu, “Gpublocks: Gui programming tool for cuda and opencl,” *Journal of Signal Processing Systems*, p. 1, July 2018.

[4] T. Odaker, D. Kranzlmueller, and J. Volkert, “Gpu-accelerated triangle mesh simplification using parallel vertex removal,” *International Journal of Computer and Information Engineering*, vol. 10, no. 1, pp. 160 – 166, 2016.

[5] Khronos OpenCL Working Group, *The OpenCL Specification, Version Version V2.2-11*, 2019.

[6] J. Valero, *Development of an Alternative Method for Multispectral Image Segmentation based on Cartesian Complexes and their associated Oriented Matroids*. phdthesis, Universidad Distrital Francisco Jose de Caldas, Feb. 2019.

[7] V. A. Kovalevsky, “Geometry of locally finite spaces,” *International Journal of Shape Modeling*, vol. 14, no. 02, pp. 231–232, 2008.

[8] H. Whitney, “On the abstract properties of linear dependence,” *American Journal of Mathematics*, vol. 57, pp. 509–533, 1935.

[9] J. G. Oxley, *Matroid Theory (Oxford Graduate Texts in Mathematics)*. New York, NY, USA: Oxford University Press, Inc., 2006.

[10] K. Fukuda, “Lecture notes on oriented matroids and geometric computation,” tech. rep., 2004. RO-2004.0621, course of Doctoral school in Discrete System Optimization, EPFL 2004.

[11] B. Catanzaro, B. Su, N. Sundaram, Y. Lee, M. Murphy, and K. Keutzer, “Efficient, high-quality image contour detection,” in *2009 IEEE 12th International Conference on Computer Vision*, pp. 2381–2388, Sept 2009.

[12] J. López-Fandiño, D. B. Heras, F. Argüello, and M. Dalla Mura, “Gpu framework for change detection in multitemporal hyperspectral images,” *International Journal of Parallel Programming*, p. 1, Dec. 2017.

[13] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, “Contour detection and hierarchical image segmentation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, pp. 898–916, May 2011.

[14] M. Worboys and M. Duckham, *GIS: A Computing Perspective, 2Nd Edition*. Boca Raton, FL, USA: CRC Press, Inc., 2004.

[15] NVIDIA, *NVIDIA CUDA C Programming Guide (Version 10.1)*, 2019.

[16] Eclipse, *The Platform for Open Innovation and Collaboration*. Eclipse foundation, version 2019-09 ed., Sept. 2109.

[17] M. Hutter, *Java bindings for CUDA*, 2019.

[18] GDAL Development Team, *GDAL - Geospatial Data Abstraction Library, Version 2.1.0*. Open Source Geospatial Foundation, 2016.

Epsilon-Rotation Invariance using Approximate Euclidean Spheres Packing Algorithm for Cancer Treatment Planning

Anod Alhazmi
Department of Computer Science
University of Colorado
Colorado Springs
Colorado, USA
aalhazmi@uccs.edu

Sudhanshu Kumar Semwal
Department of Computer Science
University of Colorado
Colorado Springs
Colorado, USA
ssemwal@uccs.edu

ABSTRACT

Cancer treatment planning using SRS (Stereotactic Radio Surgery) uses approximate sphere packing algorithms by guiding multiple beams to treat a set of spherical cancerous regions. Usually volume data from CT/MRI scans is used to identify the cancerous region as set of voxels. Computationally optimal Sphere Packing is proven NP-Complete. So usually *approximate* sphere packing algorithms are used to find a set of non-intersecting spheres inside the region of interest (ROI). We implemented a greedy strategy where largest Euclidean spheres are found using distance transformation algorithm. The voxels inside of the largest Euclidean sphere are then subtracted from the ROI, and the next Euclidean sphere is found again from the subtracted volume. The process continues *iteratively* until we find the desired coverage. In this paper, our goal is to analyze the rotational invariance properties of resulting sphere-packing when the shape of the ROI is rotated. If our sphere packing algorithm generate spheres of identical radius before and after the rotation, then our algorithm could also be used for matching and tracking similar shapes across data sets of multiple patients. In this paper, we describe unique shape descriptors to show that our sphere packing algorithm has high degree of rotation invariance within ±epsilon. We estimate the value of epsilon in the data set for 30 patients by implementing these ideas using Slicer3D™ platform.

Keywords
Rotation Invariant, Slicer3D, Sphere Packing, Distance Transformation, Stereotactic.

1. INTRODUCTION

In Stereotactic radio surgery, tumors are irradiated by beams of high-energy waves. It is a challenge during cancer treatment planning to provide minimal damage to healthy tissue around the tumors that get exposed to the radiation and still radiate cancerous cells. Our goal using sphere packing is to arrange beams on "spheres" in a way that hit the unhealthy tissue and leave the healthy tissue intact. A key geometric problem in Stereotactic radio surgery planning is to fill a 3D irregular-tumor shape (ROI) with disjointed spheresIn several medical applications such as inspection of tumor or interacting with portion of a 3D volume data, the cancer, represented as Region of Interest (ROI), could be rotated at arbitrary angles. If a sphere packing algorithm is used before and after such rotation, then rotational invariance suggests that there might be high correlation between spheres found by our sphere packing algorithm before and after the rotation. Defining correspondences between the original and rotated ROIs is an important task that could be solved by spheres' descriptors. If these descriptors are highly correlated, then we can anticipate that the ROIs might be similar as well. Li et al. (Li & Simske, 2002) stated that translation and scaling are easy compared to rotation. Rotation of a 3D volume data or 3D image involves simultaneous manipulation of three coordinates to maintain invariance. In the case of sphere packing, as we capture the ROI with non-

https://doi.org/10.24132/CSRN.2020.3001.4

intersecting spheres, the rotation invariance means that set of spheres will remain identical in size although their placement is expected to change under an arbitrary rotation. There are three major techniques to prove the rotation invariance: landmarking, rotation invariant features/shape extraction descriptor (Kazhdan and Funkhouser, 2003) and brute force rotation alignment mapping the problem to sub-graph isomorphism (Johnson and Garey et.al.) which is NP-Complete. *The landmarking* is normally carried out by following two methods, domain specific landmarking and generic landmarking (Szeptycki, Ardabilian, & Chen, 2009). The domain specific landmarking accepts some fixed point in the image and does rotation with respect to that about an arbitrary axis. The generic landmarking method on the other hand, finds the major axes of the 3D/2D image and that can rotate the volume or image as a whole in carrying out the rotation. Because the size of the volume data can be typically large based on the size of the data, both these approaches require that large memory storage is available as the complete voxel information is required, and usually is time consuming. *The brute force alignment* method divides/degrades the object into large number of smaller parts and works with them for rotation. This method is time consuming, complex and complicated because *parts* have to be organized. The developed code for a particular shape in this method may only apply to the data in hand and may not be generalizable. Finally, *Invariant feature/shape descriptor* involves identification of certain invariant features (measurable quantities) that remains unaltered under rotations of the 3D image or volume data.

The field of 3D object shape-similarity has been studied for years. Shape analysis is the key problem in the object similarity issues. There are different methods used for recognition of the representation of the objects such as using the global features, local features, or spatial features. The human notation of similarity is based on the overall shape which rely on the global features of the whole object. Our algorithm uses global method for capturing the whole object by filling it with spheres and then using the local method for the sphere packing based radius and distance matching, under the constraint of preserving the relative features such as size. Figure 2(b) shows the main idea of ESP algorithm where we use distance transformation isolate largest spherical volume first in the given volume data. The center and radius of this largest so far spherical volume is recorded. More details of O(n) algorithm to find distance transformation and the largest sphere are explained elsewhere (Alhazmi and Semwal, 2018a; Anhazmi and Semwal, 2018b; Alhazmi and Semwal 2019; Alhazmi 2019). Once one sphere is found, volume data is subtracted (Figure 2(c)) and the process is repeated again, and the iteration continues until we have the desired coverage or size of the sphere is very small (defined by 1-2 voxels). The output of our ESP (Euclidean Sphere Packing) algorithm is set of spheres' center and radius. This descriptor set is defined as shape-descriptor for the given volume data. The sphere-packing descriptors (SPDs) are shown in Figure 1.

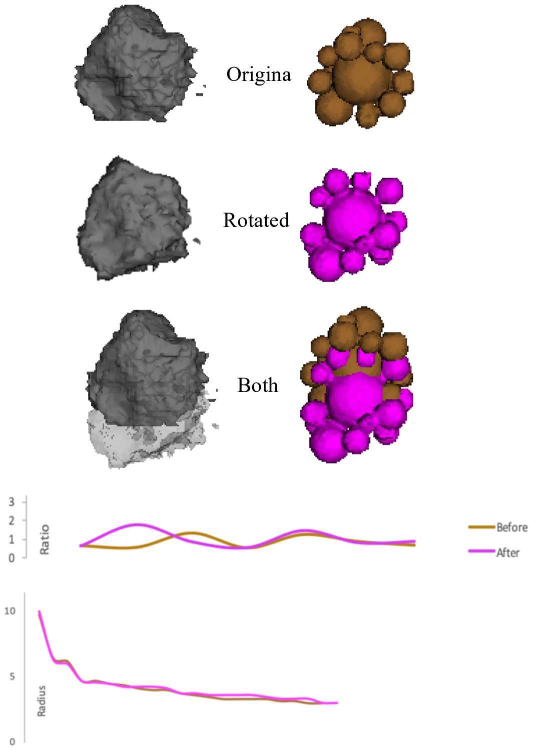


Figure 1: Top: 3D original and rotated volume with its spheres packing. Bottom: small difference in the distance ratio and radiuses measurements.

The sphere packing centers define a spatial 3D-template of the shape of the 3D object. The template is defined using the centers of all spheres generated using ESP whose main idea has been explained in the previous section. The set of centers of these spheres define vertices, and edges are defined by the sequence of edges connecting two consecutively generated spheres' centers. In summary, vertices represent the spheres centers and the edges connects every pair of the spheres. Our study focuses on analyzing this template representation under translation, scaling and rotation. Our objective to analyze the sphered description resulting from packing the 3D objects (tumors) based on *spheres packing descriptor (SPD)* that doesn't change when transforming the objects. Our proposed SPD development is a novel featureless method. SPD can describe volumetric shapes more succinctly than its voxel representation and approximated the volume-data representation by packing spheres to represent the volume data. Figure

1 shows the volume data and their corresponding SPDs. Before and after graphs are within Epsilon (ε) of each other under novel rotation and ratio measures developed in this paper. Our method characterizes the 3D object similarity by the size, shape, arrangement, and proportions of its parts. The sphere packing together with the radius and center provide an integrated approach for characterization and compression of shape information. When the volume data is rotated the SPDs provide rotational variance within +/-Epsilon. The significance of our work is that we have found a value of epsilon which works across the image data in slicer3D available for our research. In the rest of the paper we will explain these results and develop the idea of SPD using ESP method.

The outcome of the ESP is defined as a set of n unequal spheres and object P of a bounding box B. Each sphere $\boldsymbol{i} \in \boldsymbol{n} = \{\mathbf{1}, \mathbf{2}, \ldots\ldots, \boldsymbol{n}\}$ is characterized by its radius $\boldsymbol{r_i}$ and center $\boldsymbol{c_i}$. The first goal of our study is to optimize the coverage of the object P such that all spheres in n are packed in the target object inside B without overlapping. The second goal, is to apply transformations on our spheres and study the invariance of SPD under translation, scaling and rotations.

We use the packed spheres to represent the 3D object as regions. Therefore, this method of representation can also be seen *region-based* descriptor since it is based on regions. The idea is to find correspondence between object elements under different transformations. The size of the regions can be controlled depending on the treatment planning collimator size as we discussed earlier.

(a) (b)

Figure 2: (a) sphere's shape =0, multiply by (b) segmented region. (c) the result of the segment region without the pixels/voxels of the first sphere.

The invariant features are indexed with a feature vector also known as shape signatures. Then, the optimal rotation can be defined by measuring model's similarities in terms of the *distance* such that the rotation invariant property would mean that these distance measures are as close to each other with certain limit before and after the rotation. There are literally many of rotation invariant features that been used in the past, including ratio of perimeter to area, fractal measures, circularity, min/max/mean curvature, and shape histograms, etc. Lin et al. (Lin, Khade, & Li, 2012) and Yankov et al. (Yankov, Keogh, Wei, Xi, & Hodges, 2008) use time series representation as a feature vector to match the 3D shapes to prove the rotation invariance. Based on our research, most of the studies have been used spherical harmonic method to map the features of objects into a unit sphere to prove the invariance under rotation (Kazhdan, Funkhouser, & Rusinkiewicz, 2003; Nina-Paravecino & Manian, 2010; Vranic, 2003). The spherical harmonic method does not always give accurate results to distinguish between models since the internal parts of the 3D shapes may not fit in same sphere. Other researchers combined the spherical harmonic with spatial geometric moments (El Mallahi, Zouhri, El-Mekkaoui, & Qjidaa, 2017; Kakarala & Mao, 2010). The most common graph method used is skeletons. The skeletons are based on medial axis. The medial axis of the 3D objects has been used as a shape descriptor in a number of researches (Iyer, Kalyanaraman, Lou, Jayanti, & Ramani, 2003; Iyer, Jayanti, Lou, Kalyanaraman, & Ramani, 2004; Liu, 2009; Lou et al., 2003; S'anchez-Cruz & Bribiesca, 2003; Sundar, Silver, Gagvani, & Dickinson, 2003). However, this method is sensitive to noise and has a heavy computationally cost.

In this paper, we considered the set of spheres as shape-descriptors and analyzed the sphere packing before and after the rotations and looked for the similarity measure. We aimed to show that set of spheres are invariant such that even if we rotate the image, the size of the spheres and center's distances are highly correlated. We used our sphere packing algorithm to pack non-intersecting spheres into the ROIs before and after rotations. As mentioned earlier, those spheres could provide invariant shape descriptor. After rotation the voxels will be populated with the new voxel orientation. Our shape descriptor provides a novel featureless method that doesn't depend on any specific feature or texture, instead is related to sphere packing generated by our sphere packing algorithm. Our method characterizes the 3D object similarity by the shape geometries of the sphere packing, and sphere's correspondence with one another and their spatial relationships. In this paper, we show that our previous work for sphere packing (Alhazmi and Semwal 2019; Alhazmi 2019) can be used to show the invariance under rotation since our algorithm can describe volumetric shapes more succinctly than voxel representation. In this work, the spheres packing together with the radiuses and centers functions provided an shape descriptor, a novel approach for characterization and compression of shape information for 3D volume and voxel data. As we will discuss later, our heuristics is ε-invariant and has an impressive result of 96% invariant under rotations.

2. SPHERE PACKING DESCRIPTOR

We use the spheres packing to represent the 3D object, so this method of representation is called *region-based* descriptor since it is based on regions. In one of our work, Sphere Packing algorithm is used based on the maximum Euclidean distance has been studied and implemented in Slicer3D using medical imaging (Alhazmi and Semwal, 2019). Sphere packing problem is heuristically solved by using Euclidean maximum distance. The solution is to find a set of non-intersecting spheres that used greedy method and can be called largest sphere first. Each sphere is characterized by its radius and center. The size of the regions can be controlled depending on the treatment planning required size. Also, in our implementation, the density of the volume coverage can be customized such as we did in (Alhazmi and Semwal, 2019; 2018), we used 50%- 90% of the density. This means 50% to 90% (or theoretically any amount up to 100%) of the ROI is covered disjoint spheres which our algorithm finds. Of course, more the coverage, more time is taken by the algorithm to find all the spheres satisfying the user selected criteria. Generally for all patients, 50% coverage takes up to 25 minutes, and 90% takes minimum of 7 hours and maximum of 72 hours.

Our algorithm for the sphere packing is defined as a set of n unequal spheres and object P of a bounding box B. The goal of this algorithm is to pack sets of disjoint spheres inside the ROI providing certain coverage. Our strategy is as follows: A uniform grid (voxelization) is used to calculate the maximum distance of each voxel to the 3D object boundary. Then, use the maximum distance to be the radius of the first sphere and the location to be the sphere center. Iteratively, we extract new spheres each time and recalculate the distances based on the following constraints: spheres must not intersect with other spheres must completely locate inside the volume, and the volume covered by spheres is maximized using greedy strategy by subtracting the volume of the largest sphere for every iteration where largest sphere is found using a distance transformation. In our technique, sphere placements are no longer on the skeleton line. Instead, the spheres are placed wherever the maximum distance value occurred inside the ROI during that iteration (Figure 2d). We applied our maximum distance sphere packing strategy algorithm successfully on many MRIs using the Slicer3D platform; a new module in Slicer3D to be used for different shape approximation purposes.

The spheres centers of the 3D object represent a spatial template as a graph. The graph is a representation of the intersection of the sphere's centers that represent vertices of the graph of all maximum distances contained inside the 3D object, and edges connected each two consecutive generated spheres (Fig. 3). Ordering of the spheres is important for example: B, C, A will give different signature graph than A, B, C.

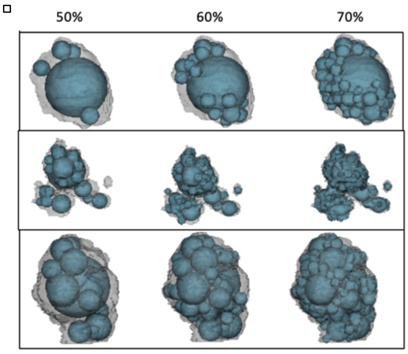


Figure 2d: 3D results of our algorithm for sphere packing in Slicer3D with 50%, 60%, 70% of packing density (gray is tumor, blue is sphere) Alhazmi and Semwal 2018; 2019).

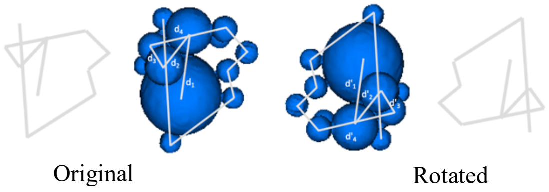


Figure 3: Spatial template graph of the original and rotated volumes generated from the intersection of the spheres' centers.

3. EPSILON ROTATION INVARIANT

We introduce a measure called epsilon-rotation invariant. Such geometric accuracy of MRI is practical especially when it used for planning radio surgery. Testing different angles of the image for beam planning is needed. Rotating the 3D volume must give the similar arrangement of sphere packing. We captured inner distances between two consecutive spheres' *centers* of our shape descriptors as an approximation to compute the difference between the two 3D shape descriptors, before and after the rotation**.** This graph distances representation is useful to abstract a geometric meaning of the 3D shape and to characterize the connectivity information. From Figure 3, assuming we rotated a 3D tumor, apart from how close d_1 is to d'_1 and d_2 is to d'_2 etc., we also look at the inner spheres' centers distances between center of the original spheres compared with the corresponding distances on the rotated volume by finding the ratio as follow:

Distance ratio =

$$\begin{cases} {}^{d_1}\!/_{d_2} = {}^{d`_1}\!/_{d`_2} & \text{invariance is met} \\ {}^{d_1}\!/_{d_2} \neq {}^{d`_1}\!/_{d`_2} & \text{invariance not met} \end{cases}$$

The inner distances between the spheres capture the distances before and after the rotation of 3D object, and find the sphere packing descriptors. In other words, we find how similar the spherical coverage is before and after, and intuitively compare that to the graph inside the spheres. Although we did not implement the orientation of such inter-distances, we expect that to be closely related for better results for our distance transformation based shape descriptors. Intuitive idea is that apart from radius being equal, the relationship between the centers should also be similar between one sphere to another. Our algorithm descriptor map entries correspond to the Euclidean distance between spheres' centers and these values are arranged in a manner that preserve the relative position of each sphere.

4. IMPLEMENTATION

We implemented our method in Slicer3D (Fig. 4). The Slicer3D (Kikinis, Pieper, & Vosburgh, 2014), is an open source medical visualization tool. The 3D slicer builds on top of different libraries such as VTK, ITK, CMake, NA-MIC, Qt and Python (Alhazmi and Semwal, 2018). Also, it contains more than a hundred modules written in C++ or Python to provide researchers many common tools and rich implementations to achieve and implement their goals. The visualization toolkit (VTK) framework is an open source with C++ libraries that contains many filters for data representation/visualization. We developed our Slicer3D Python module for sphere packing to work with the VTK for volume rotation. 3D arbitrary rotations are introduced for medical images as an extension of our previous work carried out for sphere packing (Alhazmi and Semwal, 2019). We used *VtkTransform* to apply rotation via 4x4 matrix multiplications. Our algorithm rotates images any number of degrees around x, y, and z axes. Any arbitrary rotation can be described by specifying the coordinates of the object in 3D space and rotation angels. Unlike 2D rotation, 3D rotation occurs along an arbitrary axis. Suppose the rotation angle is a, the rotations about three major axes uses well known formulae:

- Rotation along x= $\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(a) & -\sin(a) \\ 0 & \sin(a) & \cos(a) \end{bmatrix}$
- Rotation along y= $\begin{bmatrix} \cos(a) & 0 & \sin(a) \\ 0 & 1 & 0 \\ -\sin(a) & 0 & \cos(a) \end{bmatrix}$
- Rotation along z= $\begin{bmatrix} \cos(a) & -\sin(a) & 0 \\ \sin(a) & \cos(a) & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Slicer3D create a 3D scene file as Medical Reality Markup Language (MRML) and display images in physical space using patient coordinates system RAS (Right Anterior Superior), based on the information of the image spacing, origin, and direction. When applying rotation, we used the spheres packing information along with the origin and spacing. Thus, before we apply the rotation, we need to know the data of the volume:

- Position: the 3D coordinates of the object.
- Bound: the bound box of the object represented as (xmin, xmax, ymin, ymax, zmin, zmax).
- Origin: it is the position of the first voxel in the patient coordinate (0, 0, 0). It is the space origin, which is the center of all rotations
- Spacing: it is the voxels distances along each axis in the image.

Applying rotation using VtkTransform is done by following six phases as follow:

- ***Phase 1****:* Create and add a transformation node. We first create a TransformNode using VtkMRMLTransformNode, then add that node to the MRML scene. This node contains the transform ID and can store any linear transformations of composite of multiple transformations.

```
def addTransform(self):
 transformNode = slicer.mrmlScene.AddNode
            (slicer.vtkMRMLTransformNode())
```

- ***Phase 2****:* Create a homogenous 4x4 transformation matrix. VtkTransform generates 4x4 matrix that initialized to the identity matrix transformation (all zeros with ones in the diagonal) to describe the linear transformation.

```
Rotation = vtk.vtkTransform()
```

Phase 3*:* Set the parameters of rotation. Our algorithm rotates images any number of degrees around x, y, and z in z, x, y order.

```
   if rx != 0:
     rotation.RotateX(rx)
   if ry != 0:
        rotation.RotateY(ry)
    if rz != 0:
        rotation.RotateZ(rz)
```

RotateX, RotateY, and RotateZ create the rotation matrix. Since VtkTransform rotate the object around the origin (0,0,0), the rotation algorithm performs the following steps to rotate the volume about its center. The volume is first translated to its center so that its centroids lie on the center of the image instead of the origin (0,0,0). The resulting volume is then rotated according to transformation chosen by the user (x, y, and z angles values). Then, translate the volume back to its original pose.

Phase 4: Apply transformation.

```
tNode.ApplyTransformMatrix(rotation.GetMatrix())
```

Where tNode is the transform node. GetMatrix is used to return the current values to be used for the view manipulations such as rotate the current values in x, y, z angles. So, the current values (vtkMatrix 4x4) are multiplied by transformation matrix. Applying transformation is basically done by multiplying current node with the transform node and stored in simple linear transform:

```
VtkMRMLVolumeNode(tNode) * transformNode
```

Phase 5: Concatenate multiple (nested) transformations and attach volume to transform node.

```
OutputVolume.SetAndObserveTransformNodeID(tNode.GetID())
```

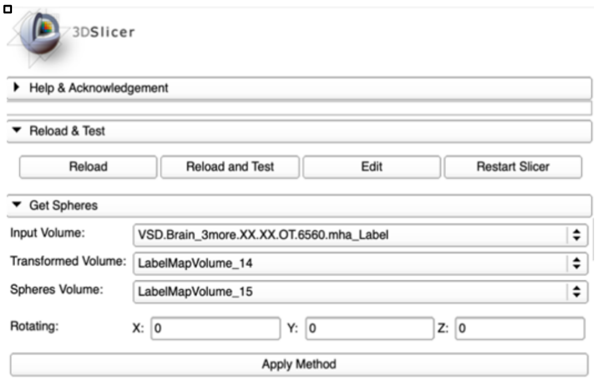
Phase 6: Harden transform.

Describe applying transformations and save it as a transformed model. Invoking transform model and *harden* transform the volume to get the correct new orientation, which will be stored in the image header. Thus, *harden* transformation is used for: changing orientation and generation of the output volume.

```
logicH = slicer.vtkSlicerTransformLogic()
logicH.hardenTransform(outputVolume)
```

Our heuristic is based on the greedy concept of generating largest spheres first, is O(n*K) where n is the number of spheres found to satisfy the chosen coverage and K is per sphere iterative constant where K= number of voxels in the volume data set. The pseudo code for Euclidean sphere packing rotation algorithm is as follow:

```
1.  Input volume as nrrd file
2.  Calculate 3D coordinates position, origin, and spacing
3.  Add a transfer node: VtkMRMLTransformNode
4.  Create a homogenous 4x4 transformation matrix: VtkTransform
5.  Set the parameters of rotation: RotateX(), RotateY(), RotateZ()
6.  Apply transformation: ApplyTransform
7.  Compute the new spacing, origin
8.  Output (Rotated Model)
9.  SegmentedRegion= Bounded rotated Model
10. Distance= EuclideanDistance(SegmentedRegion)
11. Max= maximum distance
12. $Sphere_{raduis}$= max
13. $Sphere_{center}$= location(max)     // center inside bounding box
14. $Sphere_{isocenter}$= ($Sphere_{center}$ + nrrd)    //center inside nrrd volume
15. $Sphere_{grid}$= (distance, radius, center)
16. SegmentedRegion= (1- $Sphere_{grid}$)
17. While(pixels not covered > pixels in desired coverage)do
    1)  Repeat Steps 10-16
18. Draw MultiSpheres()
```



Input Volume: VSD.Brain_3more.XX.XX.OT.6560.mha_Label
Transformed Volume: LabelMapVolume_14
Spheres Volume: LabelMapVolume_15
Rotating: X: 0 Y: 0 Z: 0

Figure 4: The GUI of our sphere packing rotation in Slicer3D.

5. Similarity Distance Ratio Calculation

The Euclidean distance from spheres' centers are compare as: $\frac{d_1}{d_2} = \frac{d_1'}{d_2'}$ where d_1 and d_2 are the distances of the original volume from center$_1$ to center$_2$ and center$_2$ to center 3. While d_1' and d_2' are the corresponding distances between the transformed volume from center$_1$ to center$_2$. By repeating the same process, we can get the full range of distances to be compared for similarity. Here is the psedo-code:

```
def getRatios(self, centerList, radiusList):
    num = int(len(radiusList) / 3)
    circleSetList = []
    distRatioList = []
    for i in range(num):
        c1 = centerList[i * 3]
        c2 = centerList[i * 3 + 1]
        c3 = centerList[i * 3 + 2]
        cList = []
        cList.append(c1)
        cList.append(c2)
        cList.append(c3)
        circleSetList.append(cList)
        d1 = np.sqrt((c1[0] - c2[0]) ** 2 + (c1[1] - c2[1]) ** 2 + (c1[2] - c2[2]) ** 2)
        d2 = np.sqrt((c3[0] - c2[0]) ** 2 + (c3[1] - c2[1]) ** 2 + (c3[2] - c2[2]) ** 2)
        distRatioList.append(d1 / d2)
    return circleSetList, distRatioList
```

5.1. Accuracy

This is the accuracy of approximating the same radiuses/ratio values after transformation. For each patient, we divide each radius or ratio in the original volume by its corresponding radius/ratio after transformation to see how well our algorithm works. Then, we take the average accuracy per patient. Finally, we find the overall average accuracy for our algorithm regarding how accurate our algorithm approximated the volume after rotation.

Accuracy = IF sphere_before < sphere_after

Sphere_before / sphere_after

else

sphere_after / sphere_before

5.2. Error Percentage

This is the percentage of the number of radiuses/ratios distances that are not covered after transformation. It is calculated by dividing the sum of the absolute difference in total number of distances ((sum_before) - (sum_after)) over correct number of distances before transformation (sum_before).

Finally, Error percentage = (Before_sum – After_sum) / sum(before_spheres)

6. RESULTS

In our experiments, we used thirty MRIs of segmented brain tumors from the BRATS dataset (Menze, Bjoern H and Jakab, 2015) separated on three datasets with ten patients on each (Fig. 5). The three datasets were already manually revised and delineated by experts board-certified neuroradiologists and radically different in size, shape and complexity, so that was helpful to our research. The tumor sizes range from 248318 to 12948 pixels.

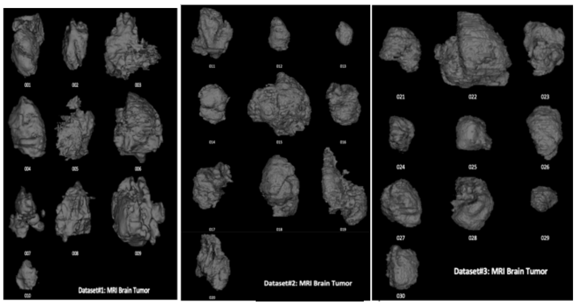

Dataset#1: MRI Brain Tumor
Dataset#2: MRI Brain Tumor
Dataset#3: MRI Brain Tumor

Figure 5: 3D MRI datasets of brain tumor.

We created arbitrary rotations to the data sets (as shown in Figure 6); and applied our algorithm of section 5. We wanted to study the ESP algorithm with the matching similarity performance of ESP algorithm before and after the rotations. That meant comparing the rations of the set of spheres which are output of our ESP algorithm before and after the arbitrary rotations. We expected a few differences as ROI before and after the rotation is expected to the voxels *slightly* differently. We first pack the ROI-volume with spheres. Then, distances ratios and radii of each sphere are calculated to be compared with the correspondence on the rotated volume of distance ratio and radii values (Section 5). More similarity measurements are investigated such calculating the accuracy along with the Mean Absolute Error (MAE) of our algorithm. Our epsilon value measure for similarity based on our study is developed using Error percentage before and after the rotations. This allowed us to manage differences that result due to the fact that voxel sizes also change and are within epsilon (ε) of each other.

Results under Epsilon (ε) -value criteria

In our study, we observed interesting patterns looking at the radius of spheres, and they are being close enough before and after the rotation. The spheres radiuses and distance ratios are actually within epsilon (ε) value criteria. The Epsilon is the maximum distance in terms of voxel size and is always given within a small range of numbers. Thus, after analyzing our 30 patients MRIs, the epsilon values of the difference spheres' radius before and after rotation are under one unit of difference, specifically within the value of 0.8 mm. Therefore, any radiuses within ± 0.8

are meant to be acceptable and there is then a high probability that the 3D volumes are similar when there are no multiple spheres with the same sphere-radius. Since previous epsilon value is based on the 50% of the packing density, we tested our epsilon value under different packing densities such as 60%, 70% and 90%. We found that, our epsilon value is consistent under ± 0.8 (Fig. 7).

On the other hand, when we consider the ratio of distance between two consecutive spheres in both before and after sphere packing list, the epsilon value between the *distance ratios* of the original volume and the rotated volume is within ± 2.5. That means, the difference in distance ratio between any consecutive spheres has to be within ± 2.5. However, increasing the packing density strongly increase this value to ± 4 in 60%, ± 12 in 70%, and ± 32 in 80% of packing density (Fig. 8). This is as expected as we go deeper in the list of spheres the difference is expected to increase as more (100+) additions are being made (See Figure 8).

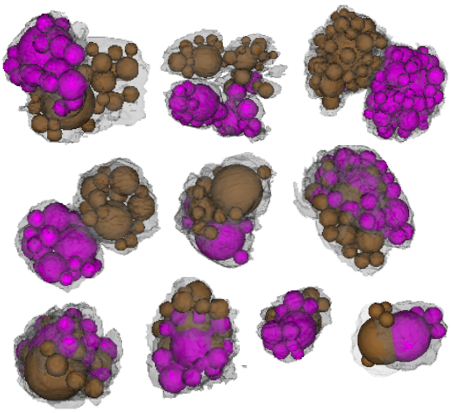


Figure 6: Before (brown) and after (purple) rotated ROIs with their ESP spheres.

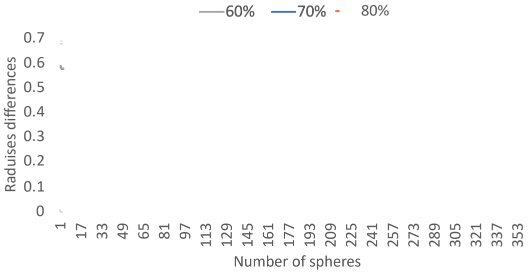


Figure 7: Radius difference plotted against number of spheres within epsilon value for three different packing density filling the ROI 60, 70, and 80%.

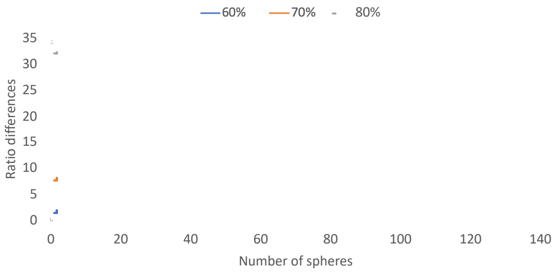


Figure 8: Increase in the epsilon value with the increase of spheres packing density from low up to 15, to medium 45, and large 100+.

The accuracy of our technique to approximate the same radiuses/ratio values after rotation is calculated for each patient data, we divided each radius or ratio in the original volume by its corresponding radius/ratio after rotation to see how well our algorithm works. The overall average radiuses accuracy percentage is 96.86% (Fig. 9). On the other hand, differences between the ratio of distance before and after, the overall Ratio average accuracy of getting same ratios is 69.23% (Fig. 10). We noted that our results are driven by a good accuracy and further 3D-spatial improvements will fetch better results.

To increase the accuracy, we further tested the accuracy of some of our results by varying the volume dimension (voxel size). Our datasets patients' grid size varies so we increased the dimensions to different values to have bigger grid size with a greater number of voxels of smaller sizes. We find that the decrease in voxel size change the accuracy level. The more the voxel size decreased (grid became finer), the accuracy increased (Fig. 11). This is expected because smaller voxel sized provide more accuracy than the bigger size

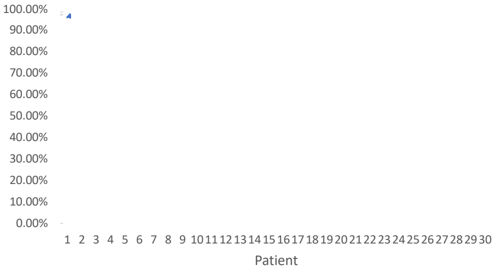


Figure 9: Radiuses accuracy.

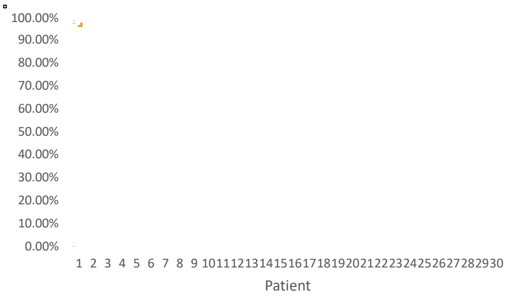


Figure 10: Distance ratio accuracy.

Still, we analyzed the data further. For each patient, we computed the absolute error between the original and the rotated radiuses/ratios. Absolute values were estimated across a range of different patients.

Absolute error = (Before_radius – After_radius).

Then, the mean absolute error (MAE) was calculated for patients using the distribution of ratios and radiuses in 30 patients. The closer this value to the zero indicate the great algorithm approximation to cover the targeted object. The overall MAE of our algorithm is 0.2. In medical applications (Irwig, 2007), we believe that this measure could be important because the absolute error represents the risk of developing recurrent disease because this value indicates the untreated cells/voxels. Being able to differentiate between patient with highest and lowest absolute risk of recurrence is an important task in order to diagnose the patient with the appropriate treatment. Therefore, the MAE could play an important role to differentiate whom radiotherapy can yield to meaningful benefits.

7. Conclusions and Future Research

The spheres radius works the best for our study for finding the similarity after rotation. Even though there are differences between the total number of calculated distances before and after rotation, our algorithm accuracy is reasonably high because it is able to calculate *almost* similar radiuses each time within epsilon value. The consistency of spheres radius is because our algorithm at each iteration finds the maximum radius distance to pick first, so increasing the number of packed spheres to cover the required voxels based on the desired packing density doesn't affect the epsilon value. Changing the topology due to equal spheres is the main reason of the increase of the epsilon value of the distance ratios because ESP algorithm can generate spheres with same radius yet distributed far away from each other, and the list of spheres generated before and after the rotations can have different *order* for these same-radius spheres in their list . The algorithm decision of choosing which sphere of the same size to place first, is the big issue here. Therefore, increasing the number of packed spheres will significantly increase the changes in topology which will results in increasing the epsilon value. In other words, when the radiuses are equal, the descriptor graph before and after might change considerably based on which sphere our algorithm suggests. In this case, aggregate we will need to collect all those spheres which are equal and replace them by the average of the center of the sphere in the shape descriptor, and so then (ε) value will be similar in the shape description before and after the rotation. Thus, set of spheres whose radius are equal are replaced with one sphere. That is expected to reduce the epsilon (ε) value further. Moreover, the topology changing in our study affect our accuracy results. We believe that eliminating equal spheres by using the enclosing sphere in our implementation will decrease the distance ratio results comparing the shape descriptors before and after the rotation of the ROI. That can be tried in future.

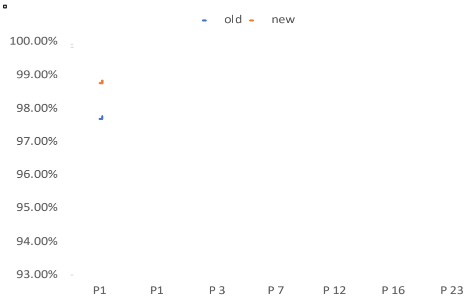


Figure 11: Increased radius accuracy.

Our novel medical visualization techniques promise to improve the efficiency, diagnostic quality and the treatment. The field of 3D shape approximation and similarity have been a focus in the area of geometry in general for several hundred years now. Shape analysis for feature extraction is the key problem in the shape approximation and similarity issues. The best way for similarity matching is to identify certain shape signatures (prominent features in the image). These signatures are then compared between the transformed images through similarity assessment, distance computation or any other appropriate methods. This paper presented a method for defining a possible invariant shape descriptor from 3D-image or 3D-volume data to be used to match the objects from different rotations/viewpoints. Our method can be applied to a wide variety of data types such as 2D images and even polygonal meshes. Our heuristics is ε-invariant and has an impressive result of 96% invariant under rotations. The experimental results prove the effectiveness of our novel idea. The proposed system was fully software implemented in Slicer3D and has been tested on 30 patient's databases. For future works, we will apply other measures such as 3D-spatial sorting based on the spheres found, or identifying a minimal volume enclosing sphere surrounding all spheres of equal radius (as mentioned earlier) to improve epsilon (ε) value further. Moreover, as Slicer3D is experimental, not FDA approved, yet used worldwide, our plan is to upload our implementations under BSD license so that world-wide communities can try the system and provide more feedback using their 3D volume data and reporting ε-value for their data.

8. ACKNOWLEDGMENTS

Thanks to anonymous WSCG reviewers for providing very helpful comments on our paper which made our paper better. Thank you.

9. REFERENCES

Alhazmi, A. and Semwal, SK. "Implementation of linear time exact euclidean distance transform in 3D slicer," *Journal of Computing Sciences in Colleges,* vol. 34, *(2),* pp. 117-124, 2018a.

Alhazmi A, Semwal, SK. ESP: Euclidean Spheres Packing Implementation for Volume Data Using Slicer3D, IEEE BHI 2019 International Conference on Biomedical and Health Informatics, Poster presentation, Chicago, 2019 (2019).

Alhazmi, A., Semwal, S. K. MRI Brain Tumor Segmentation. In *Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV)* (pp. 53-59). The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2018b.

Alhazmi, A., Euclidean Spheres Packing Transforation on Volume Data using Slicer3D, Ph.D. Thesis, Department of Computer Science, University of Colorado Colorado Springs, pp. 1-191. Thesis Advisor: Dr. SK Semwal, (2019).

El Mallahi, M., Zouhri, A., El-Mekkaoui, J., & Qjidaa, H. (2017). Three dimensional radial tchebichef moment invariants for volumetric image recognition. *Pattern Recognition and Image Analysis, 27*(4), 810-824.

Irwig, L. (2007). *Smart health choices: Making sense of health advice* Judy Irwig.

Iyer, N., Jayanti, S., Lou, K., Kalyanaraman, Y., & Ramani, K. (2004). A multi-scale hierarchical 3D shape representation for similar shape retrieval. *Proceedings of the TMCE, Lausanne, Switzerland,* , 1117-1118.

Iyer, N., Kalyanaraman, Y., Lou, K., Jayanti, S., & Ramani, K. (2003). A Reconfigurable 3D Engineering Shape Search System: Part I—Shape Representation. *ASME 2003 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference,* pp. 89-98.

Johnson, DS, Garey, M. Computers and Intractability. *A Guide to Therory of NP-Completeness,* pp. 1-389. Book. WH Freeman and Co. (1979).

Kakarala, R., & Mao, D. (2010). A theory of phase-sensitive rotation invariance with spherical harmonic and moment-based representations. *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on,* pp. 105-112.

Kazhdan, M., Funkhouser, T., & Rusinkiewicz, S. (2003). Rotation invariant spherical harmonic representation of 3 d shape descriptors. *Symposium on geometry processing, , 6.* pp. 156-164.

Kikinis, R., Pieper, S. D., & Vosburgh, K. G. (2014). 3D slicer: A platform for subject-specific image analysis, visualization, and clinical support. *Intraoperative imaging and image-guided therapy* (pp. 277-289) Springer.

Li, D., & Simske, S. (2002). Shape retrieval based on distance ratio distribution. *HP Laboratories, Palo Alto, CA, USA, Tech.Rep.HPL-2002-251,*

Lin, J., Khade, R., & Li, Y. (2012). Rotation-invariant similarity in time series using bag-of-patterns representation. *Journal of Intelligent Information Systems, 39*(2), 287-315.

Liu, L. (2009). 3d thinning on cell complexes for computing curve and surface skeletons. Washington University).

Lou, K., Jayanti, S., Iyer, N., Kalyanaraman, Y., Prabhakar, S., & Ramani, K. (2003). A reconfigurable 3D engineering shape search system: Part ii—Database indexing, retrieval, and clustering. *ASME 2003 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference,* pp. 169-178.

Menze, Bjoern H and Jakab. (2015). The multimodal brain tumor image segmentation benchmark (BRATS). *IEEE Transactions on Medical Imaging, 34*(10), 1993.

Nina-Paravecino, F., & Manian, V. (2010). Spherical harmonics as a shape descriptor for hyperspectral image classification. *Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery XVI, , 7695.* pp. 76951E.

S\'anchez-Cruz, H., & Bribiesca, E. (2003). A method of optimum transformation of 3D objects used as a measure of shape dissimilarity. *Image and Vision Computing, 21*(12), 1027-1036.

Sundar, H., Silver, D., Gagvani, N., & Dickinson, S. (2003). Skeleton based shape matching and retrieval. *Shape Modeling International, 2003,* pp. 130-139.

Szeptycki, P., Ardabilian, M., & Chen, L. (2009). A coarse-to-fine curvature analysis-based rotation invariant 3D face landmarking. *Biometrics: Theory, Applications, and Systems, 2009. BTAS'09. IEEE 3rd International Conference on,* pp. 1-6.

Vranic, D. V. (2003). An improvement of rotation invariant 3D-shape based on functions on concentric spheres. *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on, , 3.* pp. -.

Yankov, D., Keogh, E., Wei, L., Xi, X., & Hodges, W. (2008). Fast best-match shape searching in rotation-invariant metric spaces. *IEEE Transactions on Multimedia, 10*(2), 230-239.

Hybrid Approach for Orientation-Estimation of Rotating Humans in Video Frames Acquired by Stationary Monocular Camera

David Baumgartner
Christoph Praschl
RG Advanced Information Systems and Technology, Research and Development GmbH, University of Applied Sciences Upper Austria
Softwarepark 11
4232 Hagenberg, Austria
first.last@fh-hagenberg.at

Tobias Zucali
amb-technology.ai,
AMB GmbH
Hafenstraße 47-51
4020 Linz, Austria
t.zucali@amb-technology.ai

Gerald A. Zwettler
RG Advanced Information Systems and Technology, Department of Software Engineering, University of Applied Sciences Upper Austria
Softwarepark 11
4232 Hagenberg, Austria
gerald.zwettler@fh-hagenberg.at

ABSTRACT

The precise orientation-estimation of humans relative to the pose of a monocular camera system is a challenging task due to the general aspects of camera calibration and the deformable nature of a human body in motion. Thus, novel approaches of Deep Learning for precise object pose-estimation in robotics are hard to adapt to human body analysis. In this work, a hybrid approach for the accurate estimation of a human body rotation relative to a camera system is presented, thereby significantly improving results derived from poseNet by applying analysis of optical flow in a frame to frame comparison. The human body in-place rotating in T-pose is thereby aligned in the center, applying object tracking methods to compensate for translations of the body movement. After 2D skeleton extraction, the optical flow is calculated for a region of interest (ROI) area aligned relative to the vertical skeleton joint representing the spine and compared frame by frame. To evaluate the eligibility of the clothing as a fundament for good feature, the local pixel homogeneity is taken into consideration to restrict the optical flow to heterogeneous regions with distinctive features like imprint patterns, buttons or buckles besides local illumination changes. Based on the mean optical flow with a coarse approximation of the axial body shape as ellipsis, an accuracy between 0.1° and 2.0° by a target rotation of 10° for orientation-estimation is achieved on a frame-to-frame comparison evaluated and validated on both, Computer Generated Imagery (CGI) renderings and real-world videos of people wearing clothing of varying feature appropriateness.

Keywords

Object Tracking, Orientation-Estimation, Optical Flow, Image Alignment, Human Skeleton Extraction, Human Pose Estimation, Pixel Homogeneity

1. INTRODUCTION

Accurate knowledge of the orientation of a human from a monocular video is besides skeleton analysis of highest importance to allow for 3D reconstruction of the body, e.g., for subsequent analysis of the size of a garment. Furthermore, in the field of collaborative human-computer interaction with respect to industry 4.0 or general person tracking in the surveillance field, the analysis of human pose and orientation is of high interest. Using only one static monocular camera system without any knowledge of extrinsic camera parameters, the human as the target object needs to be recorded from several views, e.g., by in-place rotating in T-pose. With a static camera system and rotating jet centric aligned objects, it is inevitable to gain accurate knowledge of the exact orientation relative to the camera system to allow for subsequent analysis and post-processing such as the 3D reconstruction of the individual body shape. The deformable and non-rigid nature of a human body in motion with all intrinsic and decoupled movements of hips, head and ankles harden the task of defining exact orientation relative to a camera system. Furthermore, a human body with its rotational-symmetric and homogeneous shape will project to very similar silhouettes in the 2D video recordings if only marginally varying the rotation

https://doi.org/10.24132/CSRN.2020.3001.5

angle making it hard to derive specific, unambiguous estimations.

2. Related Work

2.1 Action Recognition

The field of action recognition and pose classification can be seen as a predecessor of nowadays joint-based human pose estimation. For most of these approaches, it is generally assumed that human detection, i.e., removing the background, already has been achieved somehow [Ger10a], and thus the silhouette contour is available for processing. While simple local features such as the shape modeled as the histogram of oriented gradients (HOG) can be exploited to deduce the human activity, incorporation of motion as the histogram of optical flow (HOF) [Lap08a] for spatio-temporal context introduces additional robustness in video processing. Besides shape, also spatio-temporal texture features can be utilized by applying local binary patterns (LBP) [Kell08a]. Body shape as silhouette derived after background removal can be directly used as a feature vector for clustering in the vector space [Singh08a]. State of the art feature detectors such as Speed-Up Robust Features (SURF) allows deriving robust markers as input for action recognition [Ben14a]. Nowadays, action recognition is achieved using neural networks or even Deep Learning architectures [Ron16a].

2.2 Deep Learning for Human Pose Estimation

With the evolution of Deep Learning frameworks and methodologies, accurate human pose estimation is now feasible with 2D joint skeletons derived from input video frames, e.g., with DeepPose introduced by Toshev [Tos14a]. To derive the joint skeleton in 3D, it can either be derived from a single planar RGB image utilizing Deep Learning [Li14a] or from predicting the 3D skeleton by projecting assumed 3D positions back to 2D for evaluating the best match [Che16a]. A 3D skeleton can easily be derived if incorporating multiple views with particular 3D approximation to combine the results with a simple neural network [Rho18a]. If the relative orientation of the human within the frames is known, the topic of this paper, plain 2D skeletons and epipolar geometry are sufficient for an approximation of 3D skeleton joint positions [Yan98a].

2.3 Human Orientation Estimation

For the domain of human orientation-estimation, specific deep learning models can be trained [Cho16a] at validation accuracy slightly above 80%. Given the 3D skeleton joint positions, the human orientation can further be directly estimated by e.g. calculating the orthogonal vector from a plane between neck and left/right leg [Cho16a] or by assessing the hip-rotation with a plane between chest and left/right hip [Wei19a]. Likewise, the human face orientation can be approximated from relative landmark positions such as nose, mouth, eyes from 2D [Sug05a] [Gou04a] or from 3D marker positions for full pose recovery [Der17a].

While the estimation of the human body as an elastic and deformable multi-joint kinetic skeleton is hard to determine and unambiguous, for solid bodies in case of a priori known 3D shape Deep Learning has recently led to significant improvements in 3D orientation and pose estimation [Xia18a].

A similar approach using pixel motions to evaluate the rotational changes is presented in [Pra20a]. There we describe a method to determine the rotation in the context of head-mounted augmented reality (AR) devices. In contrast to the presented paper, we are using the method for inside-out orientation determination for AR devices, while this paper introduces an outside-in approach.

3. Hybrid Approach for Human Orientation-Estimation

3.1 Overview of our approach

Image data utilized in this research work is acquired with a particular camera setup. One monocular static camera system is used to record a person rotating in T-pose on a spot facilitating simplified camera calibration. The body skeleton is tracked in 2D by utilizing OpenPose pose estimation [Cao17a]. To compensate for body movement, the 2D frames are scaled and aligned according to the spline-joint of the tracked skeletons.

A rough orientation estimation is directly derived from the body skeleton evaluating the shoulder to hip plane, while significant refinement is achieved by evaluating the optical flow. Preconditioned nearly homogeneous object position and distance from the camera, the person elliptic axial cross-section is estimated from 0° and 90° views. Utilizing the modeled person cross-section and the quantitative phase shift of optical flow when comparing two frames, the relative rotation can be calculated based on the derived translation. The pixel homogeneity, via a co-occurrence matrix, allows excluding of homogeneous areas from optical flow analysis, resulting in a more stable and precise result.

3.2 MATERIAL

With a very broad field of application in terms of the used camera system, this paper differentiates between tests using Computer Generated Imagery (CGI) data and those using real-world data to address a wide range of scenarios. The first of the two used materials is shown in Figure 1 and uses the game engine Unity to render a virtual environment containing a model of a person, which rotates around its spine.

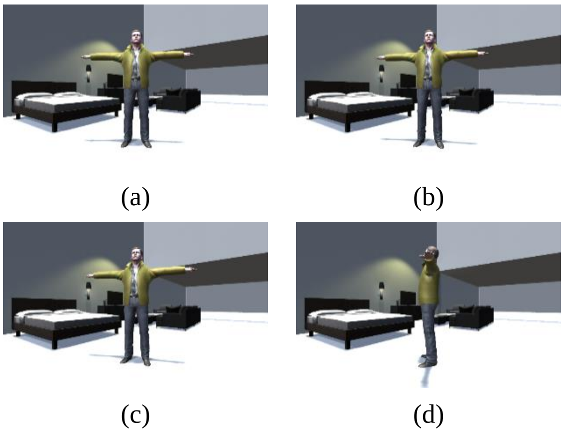


Figure 1: Sample images of a virtual person model with a rotation of (a) 0°, (b) 10°, (c) 20° and (d) 90°.

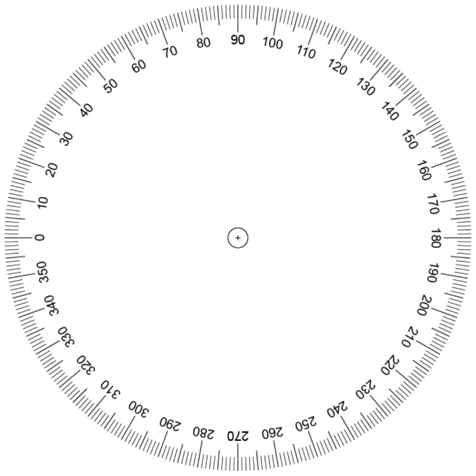


Figure 3: Degree circle underlay used as a ground truth for the real-world tests.

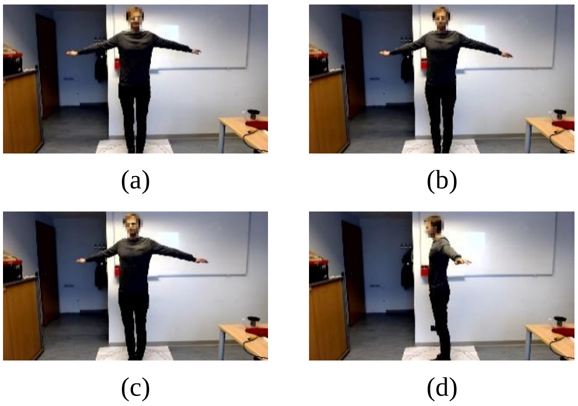


Figure 2: Sample images of a real-world person standing on the degree circle shown in Figure 3 with a rotation of (a) 0°, (b) 10°, (c) 20° and (d) 90°.

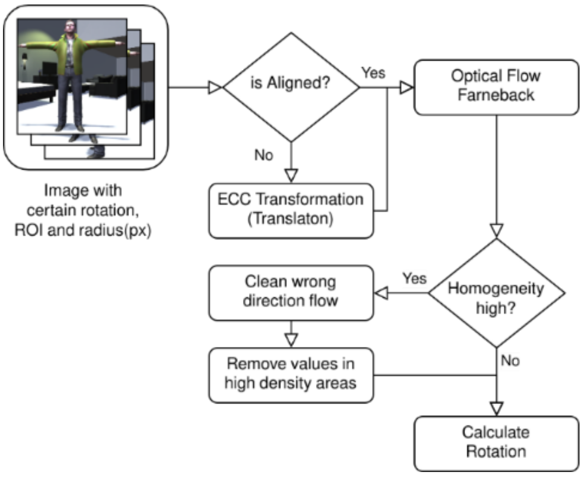


Figure 4: Shows the required overall workflow steps of the rotation calculation.

That allows us to verify the approach in a perfect setting with a precisely known rotation as the ground truth and without additional distortions related to the used camera or, e.g., movements of the camera. However, it also allows us to evaluate the effect of such influences, which can be configured/simulated in such a virtual environment.

In the second scenario, videos are captured form a person who is rotating in place around their spine, shown in Figure 2. The usage of real camera footage is associated with, among other things, the problems mentioned above, but allows us to evaluate our approach in a real-world scenario, in which it finally should find application. In this test environment, we also distinguish between two camera settings – a positional static and a dynamic one. In the first setup, we are using a fixed camera on a tripod, and in the second one, the camera is held by a second person. This additional differentiation allows comparing the synthetic scenario tests with the static camera. First, to evaluate the influence of different camera-related impacts, and secondly, the influence of real-world conditions that contain additional sources of errors, such as (minimal) movements of the camera. As the ground truth in the real-world scenarios, we use a motorized turntable that can be rotated degree-wise and is mounted on a printed degree circle, shown in Figure 3.

In both, the synthetic scenario as well as the real-world scenario, we evaluate the rotation of the person at 0°, 10°, 20°, and 90° each based on a single image frame with 24-bit color depth. The approach is tested with images of size 1920x1080px. For this, the synthetic scenario scene is exported with this resolution at the corresponding rotations. The real-world images are taken with a Logitech C920 that is mounted on a tripod in the static scenario and in the dynamic one with an iPhoneXR.

3.3 METHODOLOGY

Identifying the rotation of an object with a monocular camera requires various aspects to be checked and corrected. First of all, three frames are necessary, as described in section 3.2, one as origin, one as the target, and one with a 90° view. Handheld cameras usually introduce movement between consecutive images. The target image therefore has to be aligned to prevent such an error first. This alignment has to be a translation only and there must not be any Euclidian, affine, or homography transformation. This transformation allows us to align the background and to preserve the motion between the frames. Figure 4 shows the overall abstract workflow. It highlights the required steps for this approach to calculate the rotation of an object within images.

Following the alignment correction, the optical flow gets calculated for the entire image but evaluated only within a small region of the rotating object. Since the flow in the direction of the vertical axis is not relevant, the subsequent process steps ignore it. Within the resulting features (a two-dimensional array), the direction of the flow is essential. The main rotation direction is determined by checking for one direction that has more than 75% of all flow values in its direction. If this is not guaranteed, it is impossible to calculate the rotation from the given samples. That leads to dropping all values in the wrong direction and are therefore classified as noise by the optical flow calculation. After those steps, another check further cleans the remaining values. Within the small region of interest, the homogeneity is vital to decide for further cleaning tasks or to calculate the rotation directly. Via the co-occurrence matrix based on the grayscale values in the horizontal direction, reliable homogeneity information can be acquired. Combined with the statistical variance within the horizontal flow, it is possible to decide for further cleaning or not.

To determine the homogeneity of the texture in the ROI, a gray value co-occurrence matrix M is calculated for $range = 64$ as number of bins for 8bit unsigned input image I leading to range factor $rF = \frac{range}{256}$ with

$$M(\langle T_i \rangle, k, l) = \frac{1}{width * height * |\langle T_i \rangle|}$$

$$\sum_i^{|\langle T_i \rangle|} \sum_x^{width} \sum_y^{height} \begin{cases} 1 & \begin{array}{c} if\ floor(\frac{I(x+\Delta x_i)}{rF}) = k \\ \wedge floor(\frac{I(y+\Delta y_i)}{rF}) = l \end{array} \\ 0 & else \end{cases} \quad (1)$$

applying the translation vector $T_1(1,0)$ and $T_2(0,1)$ provided as a parameter set $\langle T_i \rangle$ with translation T_i as pair $(\Delta x_i, \Delta y_i)$. Based on co-occurrence matrix M, homogeneity is calculated as

$$homogeneity(M) = \sum_k^{range} \sum_l^{range} \frac{M(k,l)}{1+|k+l|}. \quad (2)$$

The following cleaning steps intend to remove unbalance within the remaining optical flow values. If the image has many homogenous areas, then the optical flow contains peaks within the histogram of the residual values. In order to determine if such peaks exist, the histogram is split into ten equal-width slices. Values in a slice are dropped if the slice contains more than the 85 quantiles of the number of values. That cleaning results in a more uniform distribution of all flow values. This step is valid under the assumption that the image contains many homogenous areas were no real optical flow can be calculated and would bias the further calculation steps. This allows to boost/reinforce the remaining values in their contribution to the rotation calculation.

Finally, after pre-processing, the core rotation calculation requires some additional information besides the optical flow data. It involves the width of the cross-section of the rotating object. Therefore, it is mandatory to know a priori where the rotating object and its outline resides. If the object is a can, then the cross-section is always equal, but in the case of a person, a 90° rotated view is required to extract the cross-section width. This information is used as the diameter in pixels. Reducing this value by a factor of two results in a first approximation of the radius and is further used as an adjacent leg. Flowing that is the mean value of the remaining optical flow values used as the opposite side value as shown in Figure 5 as *x Flow*. The optical flow values are further corrected by calculating a scale factor with the arccosine, dividing

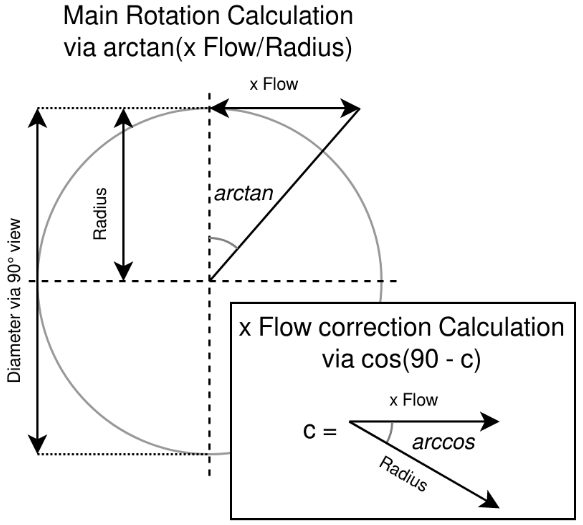
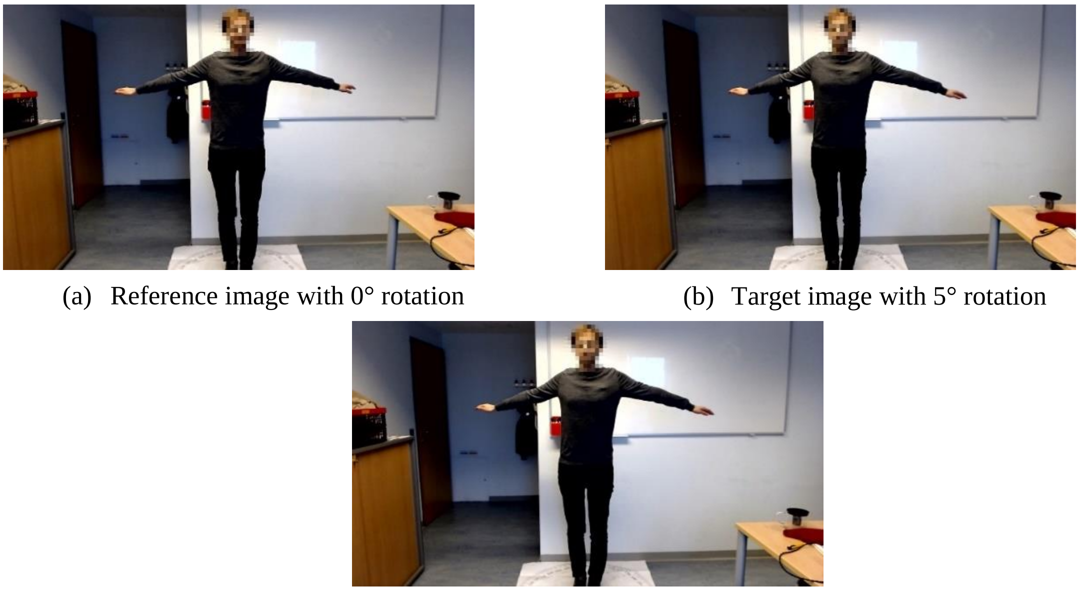

Main Rotation Calculation
via arctan(x Flow/Radius)
x Flow
Radius
arctan
Diameter via 90° view
x Flow correction Calculation
via cos(90 - c)
x Flow
c =
arccos
Radius

Figure 5: Rotation calculation via arctangent calculation with the flow values as opposite and the radius cross-section as the adjacent leg.

the values by that factor and recalculating the opposite side value. The arctangent of the fixed values represents the final rotation value.

4. IMPLEMENTATION

Our rotation estimation requires image processing steps and statistical calculations. Respective steps are described in section 3. For those tasks, we chose Python (v.3.7) as our environment to test our approach. For image processing, we rely on OpenCV (v.4.1.2.30) with the Python-Wrapper available via PyPI. Since there is no co-occurrence matrix calculation implemented in OpenCV, we additionally use Scikit-image (v.0.16.2) for calculating the matrix and deriving the homogeneity within the image from it. OpenCV additionally requires the NumPy library when used within the Python environment. NumPy (v.1.17.4) further provides the possibility for calculating quantiles and histograms. For the geometric calculation, we rely on the math implementation of the Python environment. We use the following OpenCV functions throughout the implementation: *findTransformECC, warpAffine, calcOpticalFlowFarneback*.

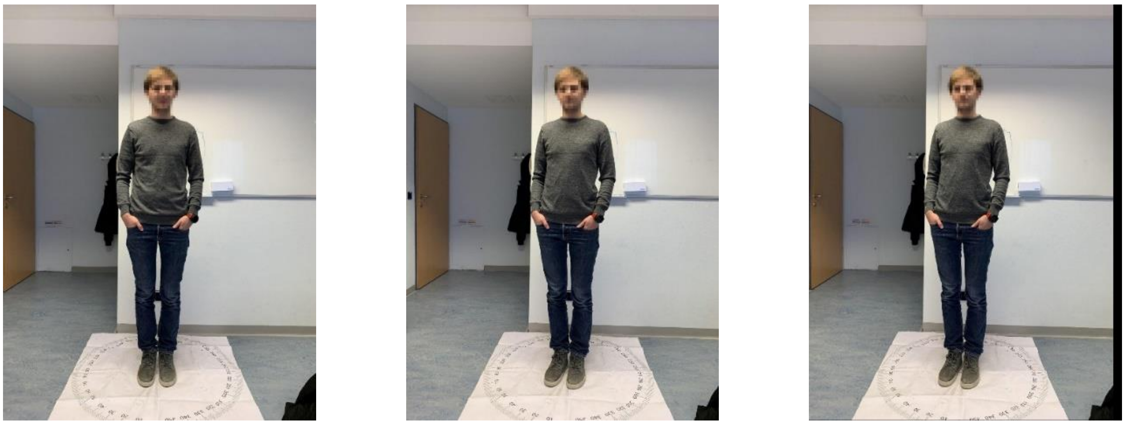


(a) Reference image with 0° rotation

(b) Target image with 5° rotation

(c) Target image with 5° rotation and alignment to the reference image, where no alignment is required due to the static mounted camera

Figure 6: Test sample with statically mounted monocular camera and no alignment required

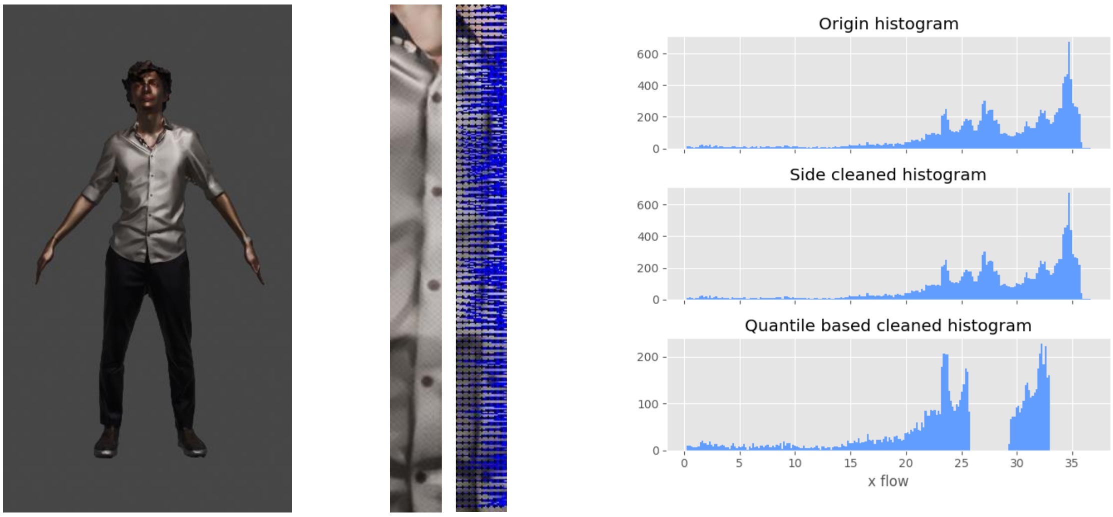


(a) Reference image with 0° rotation

(b) Target image with 10° rotation

(c) Target image with 10° rotation and alignment to the reference image

Figure 7: Test sample with handheld monocular camera and alignment required, visible as black rectangles (c) due to the alignment



(a) (b) (c)

Figure 8: (a) CGI rendering sample, (b) ROI within the rendering and the corresponding optical flow values with a rotation to the right from camera view point, (c) density of all optical flow values and the remaining densities after the cleaning

5. RESULTS

Although the person is expected to rotate in T-pose, the follow test cases cover different kinds of poses to prove the applicability of the optical-flow based orientation estimation.

5.1 Alignment Texture Normalization

Figure 6 contains a test sample with alignment. This case shows that alignment is not required because the recording happened with a statically mounted monocular camera.

The scene in Figure 7 shows, that the alignment is required in the case of cameras handheld by a second person. The aligned target image is therefore moved to the left and a bit up, visible as black rectangles in the Figure 7 (c) The movement of the hand has been corrected to calculate the flow information from the same region of interest (ROI). Otherwise, two non-aligned areas could be compared with no significant overlap of information in the ROI.

Alignment via Enhanced Correlation Coefficient (ECC) estimates the required transformation based on maximizing the correlation coefficients between the reference and target image. The best fitting alignment in our case is with translation transformation because it does not distort the content of the target image.

5.2 Tests on CGI Renderings

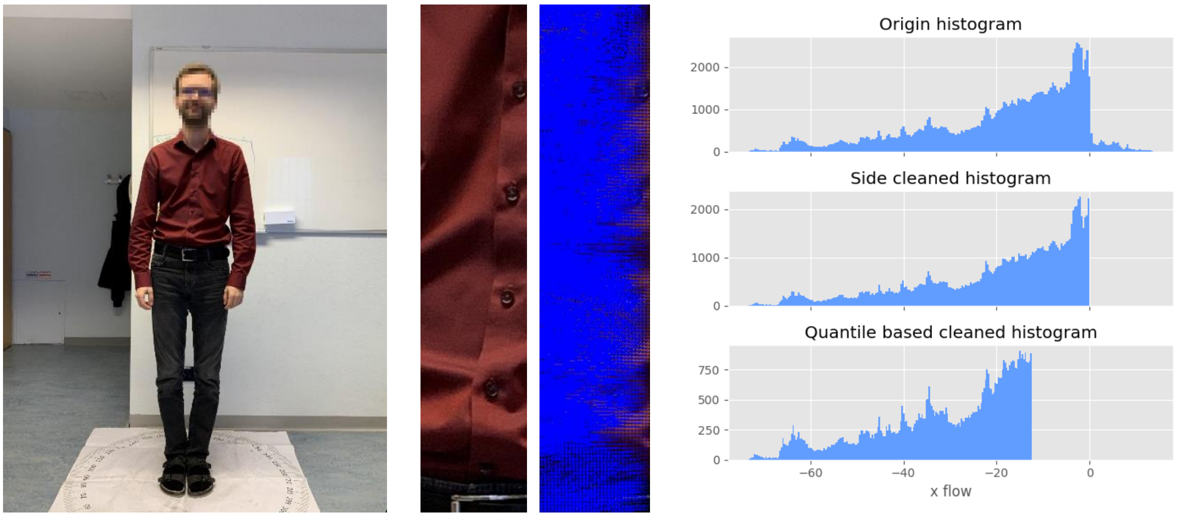
Figure 8 (a) shows the initial CGI scene. The ROI in the scene contains several features and has, therefore, a low homogeneity value of 0.267. The Figure 8 (b) contains the ROI clip out and a visualization of the optical flow. Figure 8 (c) contains the histogram of the optical flow values. It highlights a small density of noise in the range of 0 to 15. The mean is 27.97, and the median at 28.79. The calculated radius for this scene is 110. The calculation result is 14.73° with mean and 15.14° as the median value, after the correction. The expected target rotation for this sample was 15° and is consequently below the ±2° bound of the target rotation. The quantile-based cleaning, as the histogram shows in Figure 8 (c) worsens the result to a rotation value of 12.72°.

All synthetic scenario scenes results are shown in Figure 10. There are eight cases for 5°, four cases for 10° and one case for 15°. The 5° rotation are between (0°, 5°) and (5°, 10°). The 10° rotation are always between (0°, 10°). Further tests with less rotation in the range of 0° to 10° were tested with equal accurate results as the 5° target.

5.3 Tests on Real-World Recordings

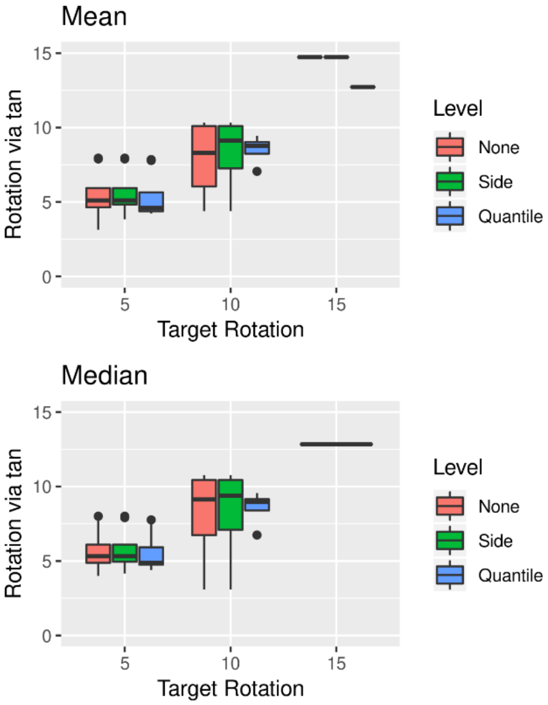
The real-world recordings achieved similar results as with CGI renderings. Figure 9 shows an example from a real recording done with a handheld camera. The homogeneity in the ROI states a value of 0.487, which concludes to apply all preprocessing cleaning steps.

Figure 9 (c) highlights the steps of the cleaning process. First, the side cleaning and finally the quantile cleaning. The target rotation in this sample is 10°, starting at 0°. Without cleaning, a rotation of 6° can directly be calculated. After both cleaning steps and correction, the final rotation results in 10.05°, which represents a very accurate result. The median flow value is, therefore, 30.71, the median value 26.92, and the radius 176. The rotation based on the median value only achieves 8.8°.



(a) (b) (c)

Figure 9: (a) real-world recording with printed degrees below, (b) ROI within the recoding and the corresponding optical flow values with a rotation to the left from camera viewpoint, (c) density of all optical flow values and the remaining density



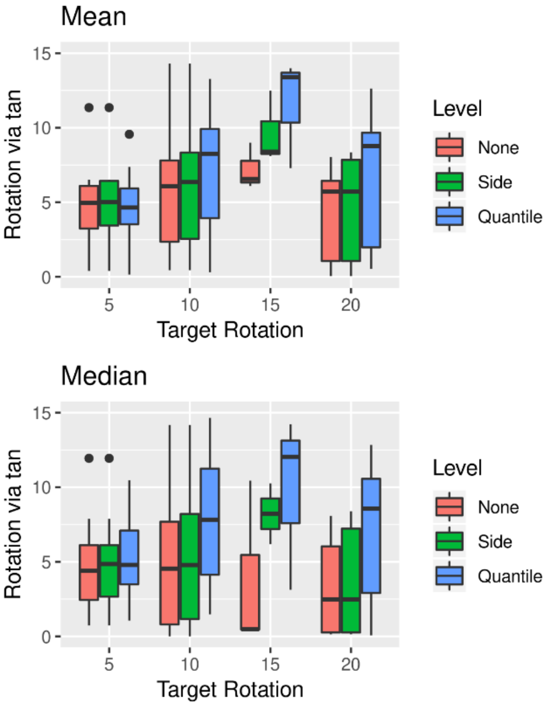


Figure 10: All synthetic scenario results with the cleaning steps and the resulting rotation by mean and median value from the optical flow.

Figure 11: All real-world results with the cleaning steps and the resulting rotation by mean and median value from the optical flow.

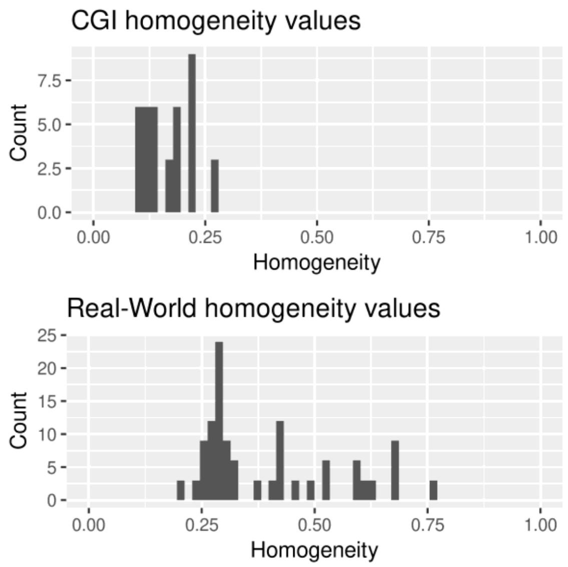


Figure 12: Homogeneity values between CGI rendering samples and real-world samples showing a significantly higher value.

All real-world results are summarized in Figure 11. The target rotation cases range from 5° to 20° with nine for 5°, 21 for 10°, three for 15° and seven for 20°. It is clear to see that with a rotation > 10° as target, the results get more incorrect. The samples itself inflict that because no feature information cannot be compensated like a too homogeneous ROI.

5.3 Homogeneity between CGI Rendering and Real-World Recordings

As expected, the homogeneity is utterly different between the CGI renderings and the real-world recordings, as visible in Figure 12. The CGI scenes usually have more features based on the perfect texture. The homogeneity values are significantly lower and centered at 0.167 and the real-world values at 0.396. In real-world recordings, it depends on the sharpness of the image, such as the focus, for example.

6. DISCUSSION AND CONCLUSION

The results show that the approach is working from controlled scenarios such as CGI rendering but also for real-world recordings. In all cases, it is visible that the rotation cannot be calculated as perfect as wanted in every scenario. Since the approach is relying on optical flow information, it is not possible to calculate a rotation higher than 45°. On the other side, the rotation calculation between contiguous frames from a video allows tracking of the rotation in a range lower than 10°. In this range, as stated, it is possible to extract the information at a high confidence.

We also tested scenarios with people wearing T-shirts with only horizontal textures. For those cases, the homogeneity is high, and by ignoring this information, the resulting rotation is far away from the desired target. That further concludes that reliable optical flow information is required. Without that, all pre and in-between cleaning steps do not automatically repair the calculated features. Therefore, it works not in every case, but with the described checks, it is possible to decide if the result is reliable or not.

Our simplified model of a person with a circle has some limitations. Naturally a person does not have a shape like this, but a more ellipse-like one instead, which has to be improved. Further, the rotation is typically not around the center of such a model. It is more likely around an asymmetric point between the center of our model and the backside. The first tests conclude that the resulting rotation can be more accurate by moving the center nearer to this desired arbitrary point.

7. OUTLOOK

Future tests will focus on the applicability of the presented orientation estimation for human shape reconstruction with real-world data. Due to the skeleton-based alignment, the presented accuracy should be sufficient with ±2° to allow for 3D human body reconstruction from a single static monocular video feed.

As the static monocular video acquires frames from one entire rotation of 360°, some potential for improvement as post-processing similar to VSLAM loop closure exists. If, e.g., the orientation evaluated from the neighboring frame pairs $(n_0, n_1), (n_1, n_2)$, …, (n_m, n_0) sums up to $360 + err$, then all frame-to-frame estimations can get scaled with $s = \frac{360}{360+err}$. Furthermore, the input video can be processed at various sampling rates, e.g., ten frames, 20 frames, and so on combining the particular results by applying maximum likelihood estimation for higher robustness.

8. ACKNOWLEDGMENTS

Our thanks to the Austrian Research Promotion Agency FFG for facilitating project TrueSize with basic funding program number 872105, fed by research budget of the Federal Republic of Austria.

9. REFERENCES

[Ben14a] Ben-Musa, A.S., Singh, S.K., and Agrawal, P. Suspicious Human Activity Recognition for Video Surveillance System. Proc. of the Int. Conf. on Control, Instrumentation, Comm. & Comp. Technologies (ICCICCT-2014), 2014.

[Cao17a] Cao, Z., Simon, T., Wei, S. E., & Sheikh, Y. Realtime multi-person 2d pose estimation using part affinity fields. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[Che16a] Chen, A.-H., and Ramanan, D. 3D Human Pose Estimation = 2D Pose Estimation + Matching. CoRR, 2016.

[Cho16a] Choi, J.C., Lee, B-J., and Zhang, B.-T. Human Body Orientation Estimation using Convolutional Neural Network. CoRR, 2016.

[Der17a] Derkach, D., Ruiz, A., and Sukno, F.M. Head Pose Estimation Based on 3-D Facial Landmarks Localization and Regression. 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017), 2017.

[Far03a] Farneback, G. Two-Frame Motion Estimation Based on Polynomial Expansion. Proce. of the 13th Scandinavian Conf. on Image Analysis, 2003.

[Ger10a] Geronimo, D., Lopez, A.M., Sappa, A.D., and Graf, T. Survey of pedestrian detection for advanced driver assistance systems. IEEE Trans Pattern Anal Mach Intell., vol 32(7), 2010.

[Gou04a] Gourier, N., Hall, D., and Crowley, J.L. Estimating Face orientation from Robust Detection of Salient Facial Structures. FG Net Workshop on Visual Observation of Deictic Gestures, 2004.

[Kell08a] Kellokumpu, V., Zhao, G., and Pietikäinen, M. Human Activity Recognition Using a Dynamic Texture Based Method. Proc. of the British Machine Conf, 2008.

[Lap08a] Laptev., I., Marszalek, M., Schmid, C., and Rozenfeld, B. Learning realistic human actions from movies. Proc. of the IEEE Comp. Soc. Conf. on Comp. Vision and Pattern Recog, 2008.

[Li14a] Li, S., and Chan, A.B. 3D Human Pose Estimation from Monocular Images with Deep Convolutional Neural Network. Asian Conf. on Comp. Vision (ACCV), 2014.

[Pra20a] Praschl, C., Krauss, O., Zwettler, G. Enabling Outdoor MR Capabilities For Head Mounted Displays: A Case Study. International Journal of Simulation and Process Modelling 2020.

[Rho18a] Rhodin, H., Salzmann, M., and Fua, P. Unsupervised Geometry-Aware Representation for 3D Human Pose Estimation. CoRR, 2018.

[Ron16a] Ronao, C.A., and Cho, S.-B. Human activity recognition with smartphone sensors using deep learning neural networks. Expert Systems with Applications, vol 59, 2016.

[Singh08a] Singh, M., Basu, A., and Mandal, M. Human Activity Recognition Based on Silhouette Directionality. IEEE Trans. on Circuits and Systems for Video Technology, 2008.

[Sug05a] Sugimoto, A., Kimura, M., and Matsuyama, T. Detecting human heads with their orientations. Electronic Letters on Computer Vision and Image Analysis 5(3), 2005.

[Tos14a] Toshev, A., and Szegedy, C. DeepPose: Human Pose Estimation via Deep Neural Networks. IEEE Conference on Computer Vision and Pattern Recognition, 2014.

[Wei19a] Wei, G., Lan, C., Zeng, W., and Chen, Z. View Invariant 3D Human Pose Estimation. CoRR, 2019.

[Xia18a] Xiang, Y., Schmidt, T., Narayanan, V., and Fox, D. PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes. CoRR, 2018.

[Yan98a] Yaniz, C., Rocha, J., and Perales, F. 3D Part Recognition Method for Human Motion Analysis. Proc. of the Int. Workshop on Modelling and Motion Capture Techniques for Virtual Environments, 1998.

Influence of Measuring Points in Ultrasonic Tests on the Tomographic Image Quality of Concrete

Marinara A. N. Moura
University of Campinas
R. Paschoal Marmo, 1888
Brazil 13484-350, Limeira, SP
m228048@dac.unicamp.br

Marco A. G. Carvalho
University of Campinas
R. Paschoal Marmo, 1888
Brazil 13484-350, Limeira, SP
magic@ft.unicamp.br

Gisleiva C. S. Ferreira
University of Campinas
R. Paschoal Marmo, 1888
Brazil 13484-350, Limeira, SP
gisleiva@ft.unicamp.br

ABSTRACT

In the process of creating digital images of concrete, wave propagation can be used to generate a suitable inner representation of this element, producing an ultrasonic tomography. Although tomography devices are efficient in generating images, they are high-cost and not very accessible. On the other hand, images obtained through ultrasound tests allow identifying the presence of flaws in different types of materials at lower costs. In ultrasonic tests, ultrasonic wave velocity paths are generated from different measurement points. Thus, in order to improve the data range, interpolation techniques can be used to estimate the acoustic velocity at the unmeasured test points, completing the image creation process. This work aims to evaluate the influence on the quality of the generated image of (1) the spatial resolution and (2) the number of measurement points in the ultrasonic test. Experiments were carried out using Inverse Distance Weighting interpolation technique and ultrasonic tomography on concrete beams with and without flaws. The resulting images indicated the importance of the appropriate choice of number and arrangement of measuring points in the ultrasonic test to properly detect internal flaws. In addition, we verify that a higher spatial resolution may also interfere in the identification of internal flaws in the image.

Keywords

Concrete, Ultrasonic Test, Ultrasonic Tomography, Data Interpolation.

1 INTRODUCTION

Concrete is the most used building material. Several structures, such as Bridges, Buildings, Structures, among others, are continuously subject to the effects that degrade their structural integrity [MEH06]. For this reason, it is necessary to take preventive actions to inspect and monitor the concrete, to ensure the safety of the structures and maintain their design performance.

It is usual to notice internal flaws in reinforced concrete construction elements, due to the execution failures or curing problems. Among the most common flaws are the concreting voids, which are difficult to identify and locate and can cause harmful consequences to structures. The difficulties on identifying and locating this flaw can be managed by using non-destructive inspection methods. The Non-Destructive Testing (NDT) is a technique used to evaluate a material without damaging it.

Among the advantages of NDT, it allows controlling the concrete quality in new constructions, evaluate the conditions of existing structures and also ensure the quality of repair work. In addition, NDT are used to assess the uniformity of the material, the modulus of elasticity and strength of concrete [CAR19].

The ultrasonic test uses the wave propagation to represent the flat section of the concrete in a 2D image. This technique has been widely used in the inspection field because it is portable, non-invasive and low-cost. In fact, the test is the most reliable, easy and portable non-destructive method used for assessing elastic properties of materials [HAS12].

Several studies have discussed the use of ultrasonic tests for detecting internal flaws in concrete. Parameters such as the number of measuring points in the test can significantly influence the results, discussed by [HAA16] and [PER19].

By means of the interpolation technique it is possible to reconstruct or approximate the continuous signals from a series of discrete points [CHE19]. This technique has been widely applied to super-resolution and im-

https://doi.org/10.24132/CSRN.2020.3001.6

age reconstruction in ultrasonic tomography [DON16] [FAN17] [FAN18].

This study intends to evaluate the influence of the number of measured points in the ultrasonic test and, consequently, on the tomography image quality generated after the interpolation process. The influence of the resolution will also be evaluated on the quality of the generated image. The interpolation method used to generate the images was the Inverse Distance Weighting (IDW).

The reminder of this paper is organized as follows: In Section 2 we address related work and basic concepts. The proposed method, the fundamentals and details of each step in the method are addressed in Section 3. The experiments and results are shown in Section 4. Finally, in Section 5 we presented our conclusions.

2 RELATED WORK AND FUNDAMENTAL CONCEPTS

This section introduces some basic concepts of Ultrasonic Tomography and Data Interpolation. Furthermore, some of related known methods are discussed.

2.1 Fundamental Concepts of Ultrasonic Tomography

Ultrasonic tomography is an NDT method that allows the mapping of the entire internal section of the object from its projections of ultrasonic waves. In contrasting with traditional ultrasonic single measurement techniques, tomographic ultrasound technique provides methods of sampling a substantial volume rather than a single point, with an automated way of data collection and processing [KOU19]. It provides visualization, either by the cross section or by the three-dimensional structure. A better identification of the anomalous regions and determination of the physical properties of the measured region can be obtained [BON00].

Recently, ultrasound tomography has attracted a great deal of interest in a wide spectrum of applications. In the industrial field, a study demonstrated the effectiveness of an online sound speed travel-time ultrasound computed tomography system to inspect liquid mixtures of different densities. The study shows the sound-speed tomographic imaging to monitor liquid elaboration processes [KOU19]. In the biological field, the development of multi-spectral optoacoustic tomography techniques was proposed by [MER19] for whole-body computed tomography of mice, in a devised a hybrid transmission-reflection optoacoustic ultrasound (TROPUS) imaging platform. The method proved that the noninvasive technique of ultrasound imaging can also provides highly complementary information on elastic and functional tissue properties of small animals, enhancing optoacoustic image quality.In additional, Although efficient tomography devices are already available, the ultrasound test is more cost-effective and can produce even better images. Regarding to the ultrasonic test, it is performed by positioning the transmitting transducer and the receiver transducer, of the ultrasonic equipment, either side of the specimen as shown in Figure 1. The transmitting sends a compression wave through the specimen, and the receiver records the full waveform on the other side. The time the wave takes to propagate through the specimen is registered, i.e., the Time Of Flight (TOF). In addition to the direct measurements, the test can be performed with the transducers on the same surface (indirect measurements) or on adjacent surfaces (semi-direct measurements), according to the American Standard ASTM-C597. [AST02].

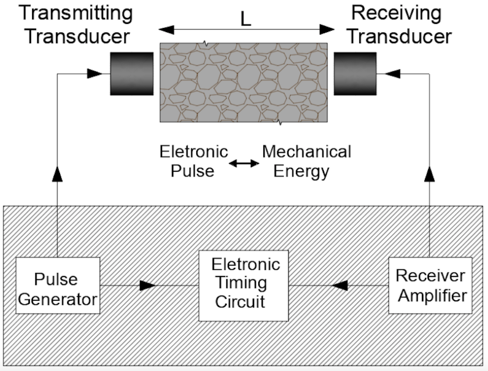


Figure 1: Ultrasonic Test on a Concrete Specimen of length L.

In the ultrasonic reconstruction of a homogeneous material, the mechanical waves propagate in straight lines between the transducers. However, in a concrete specimen, the mechanical waves may deviate from an internal flaw, affecting the resulting tomography's [PER19].

2.2 Fundamental Concepts of Data Interpolation

In an ideal interpolation, given the appropriate conditions, a continuous signal can be exactly reconstructed from the discrete points [CHE19]. A spatial interpolation can predict the values of the unknown points by processing the surrounding variables with meaningful values within the same region. This is often used when the measured data needs to be transformed from discrete points to a continuous data surface. [FEN14]

Based on the assumption that stress wave propagates as straight lines in the cross section of a tree, [FEN14] presented a study to investigate the effectiveness of using a proposed interpolation method to reconstruct a tomographic image from stress wave TOF data. The main hypothesis was that every point in the cross section of a tree trunk was related to other points and the adjacent points were more closely related than distant points.

Their results included a good match found between tomographic images and the true physical conditions of the samples.

In concrete, the use of interpolation for generation and reconstruction of images is still not widely discussed. However, some studies presents the advantage of the method applied to concrete. In the study proposed by [FAN17], the maximum likelihood expectation maximization algorithm and ultrasonic TOF data was used to reconstruct concrete images. In order to obtain a high quality image, TOF data interpolation and normalization were proposed and validated for image reconstruction system. In the study conducted by [GOM19] three spatial interpolation techniques for generating digital concrete images were used: Nearest Neighbor Interpolation, IDW and a modified IDW technique. The results showed that the presence of voids in concrete could be identified using the IDW interpolation technique.

2.3 Ultrasonic Tomography on Concrete

The ultrasonic test has been effectively used for the evaluation of several types of concrete internal flaws, such as void detection [DON16] and cracking on concrete [YAN19]. A successful example of implementation was achieved on the evaluation of structures repair by comparing the pulse velocity of the structures before and after repair [AGG11]. Regarding the identification of non-homogeneity in concrete, [SCH14] used ultrasonic tomography to determine the location and size of internal flaws in a sample of concrete and confirmed the reliability of the technique. In the experiment proposed by [FAN18], the images of concrete specimens were reconstructed by using transit time parameters interpolation and MLEM improved algorithm. The proposed image reconstruction technique was effective to improve the accuracy and the quality of image reconstruction on concrete. [HAA16] studied the influence of different number of measurements in the ultrasonic test. Using the principle of image reconstruction, their tomographic images were generated using the transformation of each TOF data as sums of partial traveltimes in all elements of a defined mesh. The Simultaneous Iterative Reconstruction Technique (SRIT) algorithm was used to solve the system of linear equations of all measurements generated. The results demonstrated that quality of the tomographic images depends on the number of the measurements and its distribution. [PER19] Also performed ultrasonic tests for detecting flaws in tomographic images of concrete. They used the Network Theory to improve the ultrasonic tomography in concrete. This theory uses a network of interconnected nodes, defining the permissible ultrasonic travel paths. By dividing the object into elements, each element was added to secondary nodes. The Dijkstra's algorithm is used to identify which travel path is the fastest. Then, for the principle of image reconstruction, the SIRT technique is also applied to solve the system of linear equations. The authors found out that the reduction of the mesh size (greater number of measuring points) provided a better image than the increment of secondary nodes (network theory proposed by the authors), proving the importance of choosing an adequate mesh for generating tomographic image by means of ultrasonic test. The use of more robust interpolation techniques have received attention in the last years. On the other hand, simpler techniques can have computational advantages if they are efficient for what is being proposed. [GOM18] used the interpolation technique to detect concrete internal flaws. They used IDW interpolation, as one of techniques proposed achieved satisfactory results in the ultrasonic tomographic images.

Considering previous work, our contribution are mainly concerned in the use of basic interpolation techniques to generate digital images for detecting internal flaws in concrete, enabling the generation of tomographic images at lower computational cost. Besides, the study of resolution in image quality is also directly linked to computational cost.Finally, the study was conducted on two reinforced concrete beams of different configurations. Although many works use a large amount of concrete specimen in their studies, the use of a concrete beam containing steel armour is more linked to practice, bringing the situation closer to reality.

3 PROPOSED APPROACH

In this section we describe the proposed approach of this work. It consists of 4 steps which are illustrated in Figure 2 and described though the remainder of this section.

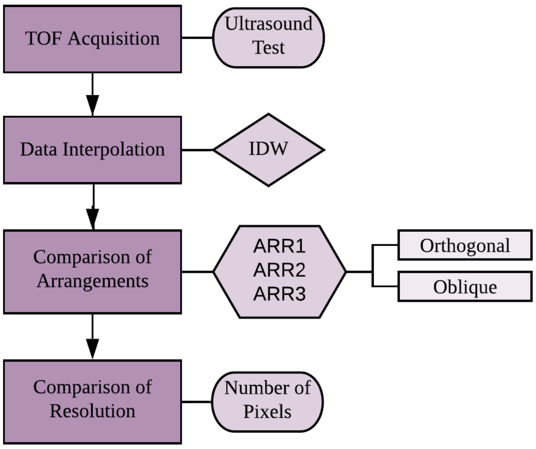


Figure 2: Workflow of proposed approach.

3.1 TOF Acquisition

The Time of Flight (TOF) acquisition consists in obtaining the ultrasonic pulse propagation times (TOF) by means of ultrasonic tests. The ultrasonic tests were performed on two reinforced concrete beams, 3 meters long, 0.4 meters high and 0.2 meters wide produced by [ROC17], both concreted with the same concrete. PVC pipes were included in one of the beams in order to simulate concreting voids defects. It had different sizes and locations according to Table 1. Figure 3(a) shows the longitudinal side of BEAM-A and the sections (1-1', 2-2', 3-3' and 4-4') used to generate the images with the TOF data. Figure 3(b) shows the longitudinal side of BEAM-B and the sections (A-A',B-B',C-C' and D-D') used to generate the images with the TOF data.

Pipe	Diameter(mm)	Length(mm)
1	25	150
2	50	200
3	25	300
4	75	100

Table 1: PVC pipes dimensions.

The comparison will be carried out between the two sections of BEAM-B that contain PVC pipes, and the other that do not contain. Besides, the 4 cross sections of BEAM-A will also be compared with the sections of BEAM-B, specially the ones without flaws since they have to be similar regardless of whether the beam has flaws or not.

3.2 Data Interpolation

The data interpolation was used to complement the generation of tomographic images. The method selected to interpolate the data in this work was the inverse distance weighting (IDW). This method is one of the most frequently used deterministic models in spatial interpolation. Its general idea is based on the assumption that the value of a point out of sample is the weighted average of known values within the neighborhood, and the weights are inversely related to the distances between the prediction location and the sampled locations [LU08].

In the IDW technique, the points closer to the point to be interpolated have more influence on the interpolation, because they have smaller distances compared to the others [HAR16].

Equation (1) describes the formulation of this interpolation method.

The value at any point P is a weighted average of values at the data point in which weighting is a function of distance to those points ([SHE68].

$$u(x) = \frac{\sum_{i=1}^{N}(di)^{p}u_{i}}{\sum_{i=1}^{N}(di)^{p}} \qquad (1)$$

Where x is an interpolated (arbitrary) point (data point). Xi is an interpolating (known) point. di is the Cartesian distance between x and xi (d[x, xi]). N is the total number of known points used in interpolation. p is a positive real number, called the power parameter. uirepresents the noise value at position i,

Since the IDW model is simple and fast, the proposed method to accomplish this study is computationally efficient.

3.3 Study of Arrangements

The third step consists in comparing the images generated with different arrangements of measuring points with orthogonal and oblique measurements. The ultrasonic test was conducted in 4 cross sections of each reinforced beam as shown Figure 3. It was used three arrangements (ARR1, ARR2 and ARR3) in order to make a comparison between the number of measured points for evaluation of the generated image. ARR1 had nine measuring points, ARR2 had five, and ARR3 had 3 measuring points. Each arrangement was done in two configurations: (1) orthogonal direct measurements, and (2) oblique direct measurements.

Orthogonal Direct Measurements: In this configuration, the TOF data was collected by using orthogonal direct measurements (Figure 3). Arrangements 1, 2 and 3 were used. Figure 4 shows the images generated in the 3 arrangements of Orthogonal Direct Measurements.

Oblique Direct Measurement: In this configuration, the test was performed in oblique direct measurements. In ARR1 it was build a mesh with nine measuring points, generating 81 travel paths. In ARR2 the mesh had 5 reading points, generating 25 travel paths. In ARR3, the mesh had 3 reading points , totaling 9 travel paths. Figure 5 shows the three stages of this configuration.

3.4 Study of Resolution

This study also includes the analysis of the spatial resolution influence on image quality for flaw detection. Considering the same size, images with better spatial resolution have a larger number of pixels than those with lower spatial resolution. Considering the size of the beam, images with the following pixel dimensions were generated: 10x20 pixels, 20x40 pixels and 40x80 pixels. If the pixel size becomes larger than the size of a flaw, it is possible that this flaw is not visualized in the digital image. Also, if it is possible to visualize the flaws with a lower resolution, the computational cost is lower. Thus, 3 different resolutions will be analyzed to choose the one that offers more advantage.

4 EXPERIMENTS AND RESULTS

The tests were conducted by the through-transmission method (direct transmissions), according to the American Report on Nondestructive testing [ACI13], with Pundit Lab commercial instrument at Proceq Company.

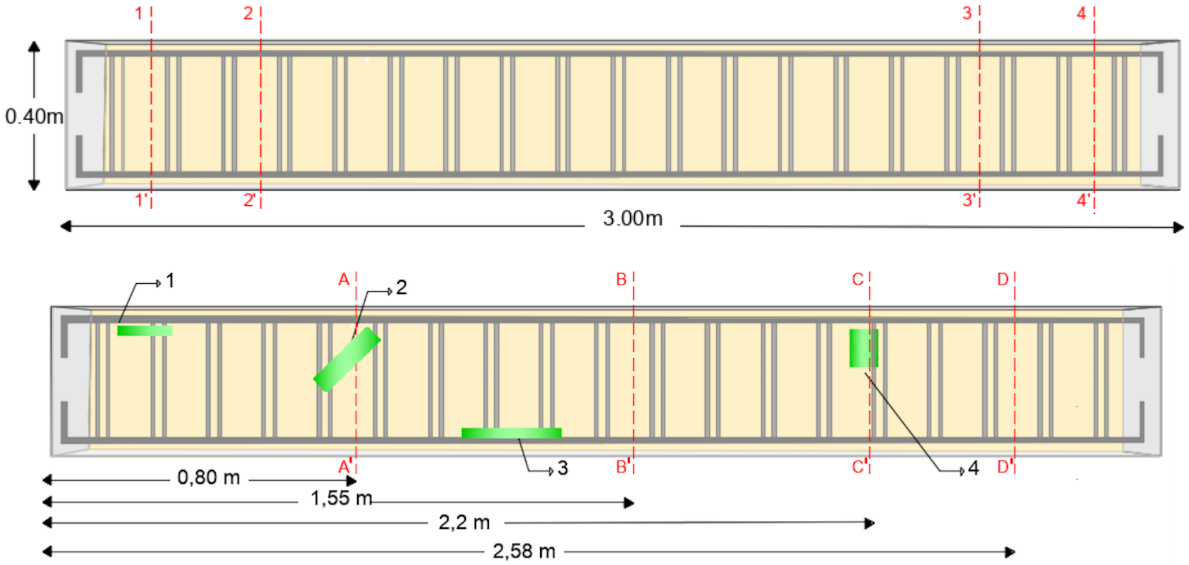


Figure 3: Reinforced Concrete Beam (a) without flaws and (b) with flaws. Adapted from [ROC17].

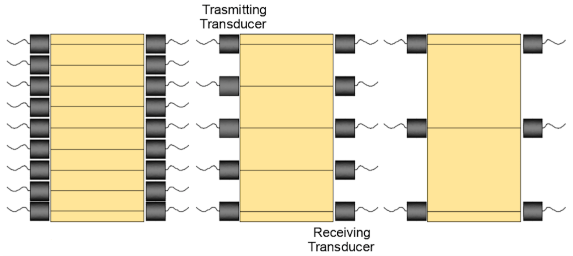


Figure 4: Cross Section of the beam in the three arrangements using Orthogonal Direct Measurements.

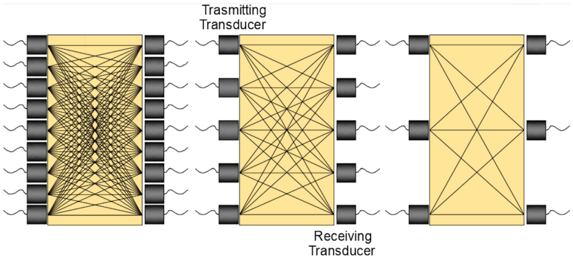


Figure 5: Cross Section of the beam in the three arrangements using Oblique Direct Measurements.

Transducers of longitudinal waves at 54kHz frequency were applied to the tests. The transducer face and the specimen were coupled by a medical ultrasound transmission gel. The images were generated using python's matplotlib library, in the "tab20c" color palette. In the IDW interpolation, the weight (P) assigned to the distance between the points used was 2.

4.1 Study of Arrangements

The images generated for the cross sections with orthogonal direct measurement are shown in Figures 6, 7 and 8, for both BEAM-A and BEAM-B. The oblique direct measurements are represented in Figures 9,10 and 11 for BEAM-A and BEAM-B.

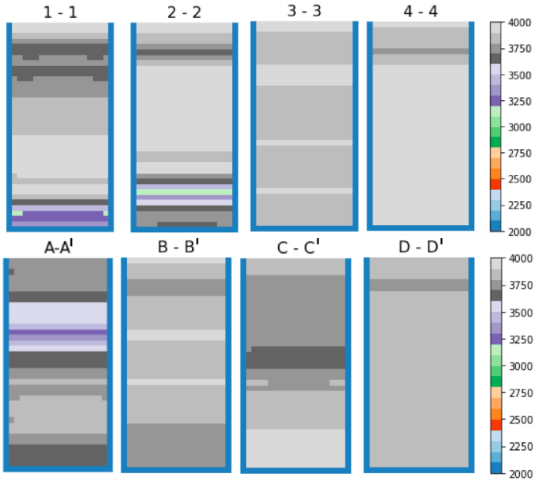


Figure 6: orthogonal direct measurements with 9 measuring points for (a) BEAM-A and (b) BEAM-B

In direct orthogonal measurements the images were generated with 1 pixel representing 1.0 cm^2. The images generated from the sections without flaws (BEAM-A, and sections B-B' and D-D' of BEAM-B) are mostly homogeneous as shown in Figures 6, 7 and 8. In this configuration, section 1-1' and 2-2' did not show consistent results, as they are more alike to sections A-A' and C-C'(sections with flaws), where the non-homogeneity may be identified as a flaw in concrete, since the measurements at some points showed lower TOF. It is suggested that the results for direct measurements might be misleading, where non-homogeneous regions (with lower TOF values) could be interpreted as flaws even in sections where they do not occur, regardless of the number of measurements points. Also, the IDW technique may impact the sections without flaws, because it calculates

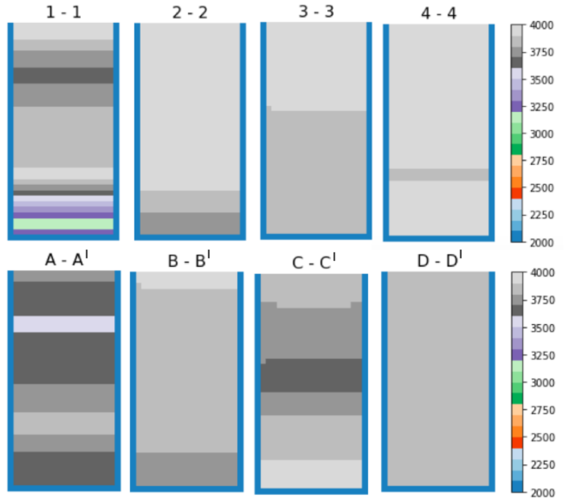


Figure 7: orthogonal direct measurements with 5 measuring points for (a) BEAM-A and (b) BEAM-B

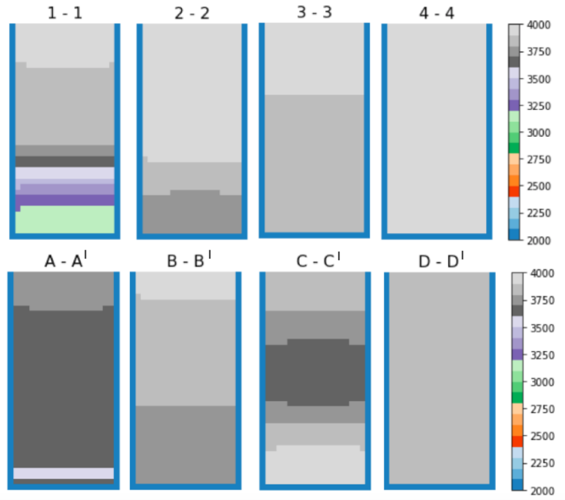


Figure 8: orthogonal direct measurements with 3 measuring points for (a) BEAM-A and (b) BEAM-B

the interpolated point considering all points measured in the section.

In direct oblique measurements the images were generated with 1 pixel representing 0.5 cm^2. The flaws had more defined shapes, represented by the colors green and orange (lower TOF) according to Figures 9,10 and 11. As expected, the measurements with more travel paths presented better results (9 measuring points). However, anomalies could be identified even with 5 or 3 arrangements of measuring points.

Different from the orthogonal measurements, in oblique measurements, sections B-B 'and D-D' of BEAM-B resemble section 1-1 'of BEAM-A. Even though these are sections without flaws, some points with lower TOF values were identified. Even so, these sections are distinguished from sections A-A 'and C-C', where most of the image actually represents non-homogeneous regions identified by lower TOF values.

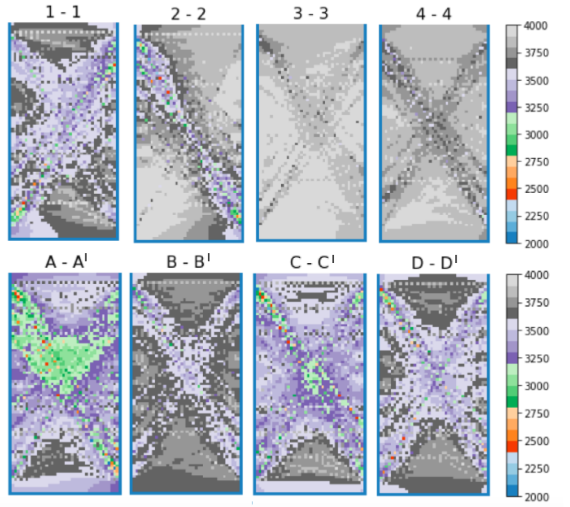


Figure 9: oblique direct measurements with 9 measuring points for (a) BEAM-A and (b) BEAM-B

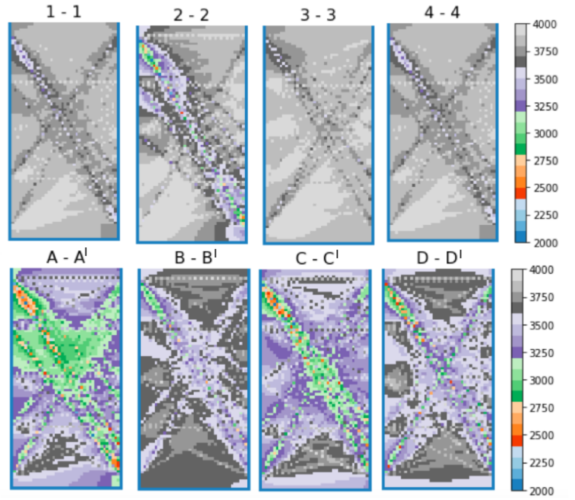


Figure 10: oblique direct measurements with 5 measuring points for (a) BEAM-A and (b) BEAM-B

The proposed algorithm only requires about 2.6 s to interpolate the images with 9 measurement points in oblique direct measurement (Figure 9), and it has much lower computational complexity than methods here commented. [HAA16] [FAN18][CHE19]

4.2 Study of Resolution

The resolution was analyzed by taking the sections with flaws (BEAM-B) as a reference, i.e., A-A' and C-C' in ARR2 (oblique direct measurements). A comparison was made among the images generated with 8x4 pixels, 20x10 pixels, 40x20 pixels and 80x40 pixels.

Figures 12 and 13 shows that the image generated with more pixels (80x40) presented better spatial resolution compared to the images generated with a smaller number of pixels (40x20 and 20x10). However, in a smaller number of pixels we understood that the presence of discontinuity could still be identified. In addition, when

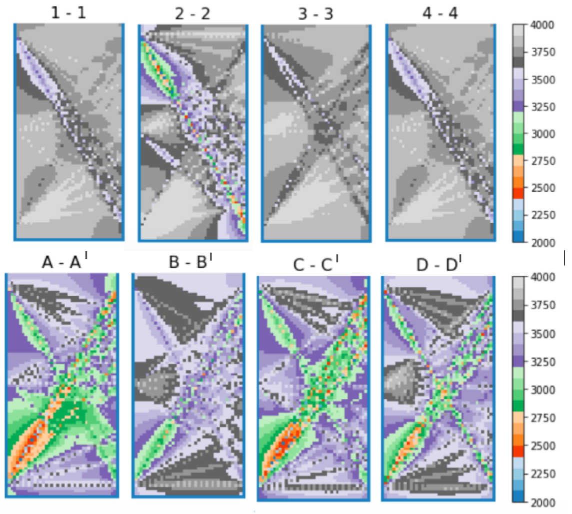


Figure 11: oblique direct measurements with 3 measuring points for (a) BEAM-A and (b) BEAM-B

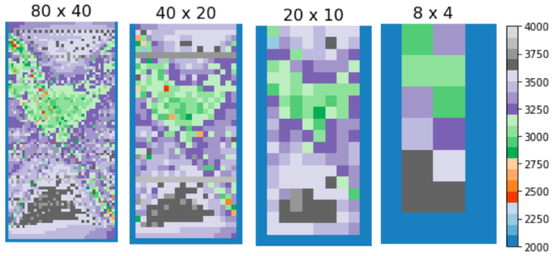


Figure 12: Image Generated with different resolutions for section A-A'.

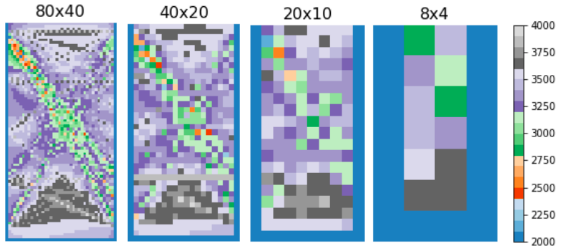


Figure 13: Image Generated with different resolutions for section C-C'.

the pixel size becomes larger than the size of the smallest beam flaw (8x4), it is hard to detect any kind of flaw.

5 CONCLUSIONS

This paper presented a study of different arrangements of measuring points in the ultrasonic test and evaluate its influence on the result of the interpolation and image quality. The IDW technique was used to interpolate the data collected in the ultrasound test. The approach provided important information regarding the experimental procedures of ultrasonic tests. Also, some limitations and favorable conditions were identified. Our initial findings indicate that the quality of the images may depend on the number of measuring points. The direct orthogonal measurements did not show consistent results, since the images with and without flaws showed similar results. However, in the oblique measurements the presence of points that could represent a flaw in concrete, was identified even in arrangements of fewer measuring points. It should be noted that the IDW technique may influence the analysis of the sections without flaws, since it considers all points of the sample. Furthermore, the spatial resolution seemed to be more relevant than the number of measuring points. The tomographic images generated with more pixels in the direct oblique measurements achieved better results. Finally, ultrasonic tomography using data interpolation presented as a satisfactory technique for assessing concrete elements homogeneity. According to results, internal flaws in concrete were find considering different analysis. However, steel reinforcement, was not considered and may induce some errors. Future work includes the use of a larger database for verifying the results and the use of other type of interpolation. In addition, we intend to use image processing metrics in order to evaluate the results quantitatively.

ACKNOWLEDGEMENTS

This work was supported by Sao Paulo Research Foundation (FAPESP), grant 2018/18520−4.

6 REFERENCES

[ACI13] ACI-228.2R. Nondestructive test methods for evaluation of concrete in structures. American Concrete Institute Report, 2013.

[AGG11] Aggelis, D., Hadjiyiangou, S., Chai, H., Momoki, S., and Shiotani, T. Longitudinal waves for evaluation of large concrete blocks after repair. NDT & E International, 44(1)pp.61-66, 2011.

[AST02] ASTM-C597. Standard Test Method for Pulse Velocity Through Concrete. American Society for Testing and Material, 2002.

[BON00] Bond, L. J., Kepler, W. F., and Frangopol, D. M. Improved assessment of mass concrete dams using acoustic travel time tomography. part i-theory. Construction and Building Materials, 14(3) pp. 133-146, 2000.

[CAR19] Carrillo, J., Ramirez, J., and Lizarazo-Marriaga, J.Modulus of elasticity and poisson's ratio of fiber-reinforced concrete in colombia from ultrasonic pulse velocities.Journal of Building Engineering, 23 pp. 18-26, 2019.

[CHE19] Cheng, D. and Kou, K. I. FFT multichannel interpolation and application to image super resolution. Signal Processing, 162:21-34, 2019.

[DON16] Dong, W., Wu, Z., Zhou, X., and Tan, Y. Experimental studies on void detection in concrete-filled steel tubes using ultrasound. Construction and Building Materials, 128:154-162, 2016.

[FAN17] Fan, H., Zhu, H., Zhao, X., Zhang, J., Wu, D., and Han,Q. Ultrasonic image reconstruction based on maximum likelihood expectation maximization for concrete structural information. Computers & Electrical Engineering, 62:293-301, 2017.

[FAN18] Fan, H., Zhu, H. Improved image reconstruction based on ultrasonic transmitted wave computerized tomography on concrete. EURASIP Journal on Image and Video Processing, 2018:129,2018.

[FEN14] Feng, H., Li, G., Fu, S., and Wang, X. Tomographic image reconstruction using an interpolation method for tree decay detection. BioResources,v. 9, n. 2, p. 3248-3263, 2014.

[GOM18] Gomes, H. G. and Goncalves, A. Analyses of Interpolation Technique for generations of TomographicImages in Concrete. Undergraduated Work (System of Information), UNICAMP (University of Campinas),Limeira, Brazil, 2018.

[GOM19] Gomes, H. G., Goncalves, A.,Carvalho, M., Moura, M., Ferreira, G., Rocha A. Analyses of Interpolation Technique for generations of Tomographic Images in Concrete. ,In Proceedings of the 61 Concrete Conferece, IBRACON ,Fortaleza, Brazil, 2019.

[HAA16] Haach, V. G. and Ramirez, F. C. Qualitative assessment of concrete by ultrasound tomography. Construction and Building Materials, 119:61-70, 2016.

[HAR16] Harman, B. I., Koseoglu, H., and Yigit, C. O. Performance evaluation of IDW, kriging and multiquadric interpolation methods in producing noise mapping: Acase study at the city of isparta, turkey. Applied Acoustics, 112:147-157, 2016.

[HAS12] Hassan, A. and Jones, S. Non-destructive testing of ultra high performance fibre reinforced concrete (uh-pfrc): A feasibility study for using ultrasonic and resonant frequency testing techniques.Construction and Building Materials, 35:361-367, 2012.

[KOU19] Koulountzios, P., Rymarczyk, T., and Soleimani, M. A Quantitative Ultrasonic Travel-Time Tomography to Investigate Liquid Elaborations in Industrial Processes. Sensors, v. 19, n. 23, p. 5117, 2019.

[LU08] Lu, G. Y. and Wong, D. W. An adaptive inverse distance weighting spatial interpolation technique. Computers & geosciences, 34(9):1044-1055,2008.

[MEH06] Mehta, P. K. and Monteiro, P. J. M. Concrete: Microstructure, Properties and Materials. McGraw-Hill,New York, 3rd edition, 2006.

[MER19] Mercep, E., Herraiz, J. L., DeÃ¡n-Ben, X. L., and Razansky, D. Transmission-reflection optoacoustic ultrasound (TROPUS) computed tomography of small animals. Light: Science & Applications, v. 8, n. 1, p. 1-12, 2019.

[PER19] Perlin, L. P. and de Andrade Pinto, R. C. Use of network theory to improve the ultrasonic tomography in concrete. Ultrasonics, 96:185-195, 2019.

[ROC17] Rocha, A. I. B. C. Identification of Pathologies in Reinforced Concrete Beams Using Ultrasound Inspection. Master Thesis (Master of Technology),UNICAMP (University of Campinas), Limeira,Brazil, 2017.

[SCH14] Schabowicz, K. Ultrasonic tomography - the latest nondestructive technique for testing concrete members - description, test methodology, application example. Archives of Civil and Mechanical Engineering, 14(2):295-303, 2014.

[SHE68] Shepard, D. (1968). A two-dimensional interpolation function for irregularly-spaced data. In Proceedings of the 1968 23rd ACM national conference, pages 517-524.ACM, 1968.

[YAN19] Yang, C. and Chen, J. Fully noncontact nonlinear ultrasonic characterization of thermal damage in concrete and correlation with microscopic evidence of material cracking. Cement and Concrete Research, 123:105797-105807, 2019.

Application of concatenable queue for parallel computational geometry algorithms

Vasyl Tereshchenko
Taras Shevchenko
National University of
Kyiv
Kyiv, Ukraine
vtereshch@gmail.com

Semen Chudakov
Taras Shevchenko
National University of
Kyiv
Kyiv, Ukraine
semen.chudakov7@gmail.com

ABSTRACT

This paper is devoted to the development of an algorithmic model that solves a set of interrelated computational geometry problems efficiently. To do this, an algorithmic environment with a unified data structure is created, which allows to implement complex use cases efficiently with respect to required computational resources. We build the environment based on the "divide and conquer" strategy. Once a convex hull is a key to a set of computational geometry problems, we offer a concatenable queue data structure to maintain it. The data structure is in the form of a modified balanced binary tree. This allows us to perform operations needed in algorithms for a set of problems in $O(\log^2 n)$ time. Furthermore we offer a way to execute the algorithms both sequentially and in parallel. In the future the algorithmic environment can be improved to support other computational models with similar properties for solving problems. As an example, the Voronoi diagram or the Delaunay triangulation can be considered.

Keywords

unified data structure, simulation problem, interrelated problems set, unified algorithmic environment, concatenable queue

1 INTRODUCTION

Nowadays, advanced computer simulations and visualizations of complex scientific researches and large-scale technical projects require to solve simultaneously a set of problems. The core of this set are problems of computational geometry and computer graphics. To solve such problems it is needed to create suitable algorithmic frameworks that would yield accurate results in real-time. Existing methods (ImageJ [Ima19a], IMARIS [Ima19b], iLastic [SSKH11]), that are based on a set of algorithms implementations organized in a package do not result in desirable efficiency and accuracy. It is worth noting that there are a lot of parallel algorithms designed to solve specifically certain computational geometry problems such as in [ACG*88, AL93, AGR94, ACG89, BSV96, Che95, CG88, GJ97, JaJ97, Rei93, Lee90]. Every such algorithm requires its computational resources and is executed independently from others. In such cases identical steps, such as preprocessing and building data structures, are repeated several times.

Therefore, an important aim in developing algorithmic models is to create a universal tool that would have a means to solve efficiently a set of problems. This tool should also execute identical steps of the algorithms once and be able to represent the results of those steps in the form of unified data structures. In [TA10] the notion of a unified algorithmic environment is introduced, which is based on the "divide-and-conquer" principle and takes into account the aforementioned features of the algorithms. In particular, preprocessing and dividing the initial set of data to form a recursion tree is common for all problems and is executed only once. During the merge stage, intermediate results are maintained in a concatenable queue for the convex hull, Delaunay triangulation and Voronoi diagram problems. This model does not repeat identical computations, which yields good performance.

In this article we first describe how the convex hull algorithm for a static set of points is decomposed into separate stages and incorporated into our unified algorithmic environment model (UAEM). Then we detailedly explain how we implement the concatenable queue, which is used in the algorithmic environment. Finally, we make a complexity analysis for the algorithm and test its performance.

https://doi.org/10.24132/CSRN.2020.3001.7

2 UNIFIED ALGORITHMIC ENVIRONMENT

In this section we described the principle of how we decompose each algorithm into distinct stages. We then use this partition to avoid repeating the computations in the algorithmic environment. The principle will be shown on a convex hull algorithm, which is similar to the one described in [OL81]. The idea there is to divide the hull into the left and right sub-hulls and represent them with two concatenable queues. This allows to achieve $O(\log^2 n)$ time for update operations such as adding or removing points from the hull. We use the approach to compute the convex hull for a static set of points. Additionally, unlike in [OL81], a convex hull here is divided into upper and lower sub-hulls.

2.1 Algorithms stages

The notion of a convex hull is simple. For a set of points S in a k-dimensional space it is the smallest convex set that comprises S. To solve such a problem means finding a subset in S that is "skeleton" for the convex hull. From now on, we will consider the case where $k = 2$.

To eliminate corner cases during the merging step we need to ensure that there are no 3 points that lie on a horizontal or vertical line. Points that violate this condition are removed from S in the preprocessing stage. Formally, the removal criterion is formulated as follows. For $a = (x_a, y_a)$ we denote $x(a) = x_a$, $y(a) = y_a$. Let the points $a_1, ..., a_k$ lie on one horizontal line and $x(a_1) < ... < x(a_k)$. Then, by the criterion, the points $a_2, a_3, ..., a_{k-1}$ must be removed. Analogously for the vertical case.

The algorithm of removing "inner" repetitions in a sorted array is trivial. To perform the preprocessing described above, it is needed to:

1. Sort points by y (if y coordinates are equal, the x coordinates are compared).
2. Delete "inner" repetitions by y coordinate using the described algorithm.
3. Sort points by x (if x coordinates are equal, the y coordinates are compared).
4. Delete "inner" repetitions by x coordinate.

As a result we get a list of points for which we can apply the recursive convex hull algorithm.

At the stage of dividing the problem into sub-problems, the list of points is split into left and right parts of roughly equal size. This can be done in $O(1)$ time for an interval $[i, j]$ by computing the average of indices i and j.

The recursion stops when there are no more than 3 points in the list. For the base case the list of points can have 2 or 3 elements. In those two cases the convex hull can be trivially constructed. The result of the base case are two concatenable queues representing upper and lower sub-hulls with one or two points in each.

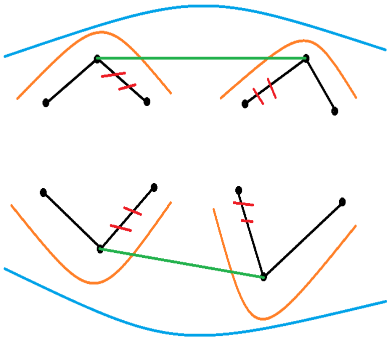


Figure 1: Merging two hulls.

To merge two convex hulls that are separable by a vertical line, we need to find upper and lower tangents that will serve as a basis for the resulting hull. Those tangents are found using the search algorithm described in [OL81]. It remains to split the sub-hulls at four found nodes that form upper and lower tangents and merge the remaining parts. An example of performing such a procedure is shown in Fig. 1.

We use the following markings on the convex hull schemes. Sub-hulls of left and right hulls are marked with orange color, sub-hulls of merged hull are marked with blue color, correct tangents are marked with green color, incorrect tangents are marked with red color and removed edges during merging are overscored with two red lines. Each convex hull is divided into the upper and lower parts due to its representation in the algorithm.

Now we will consider the corner cases that arise when performing the merging. The first of these cases is related to the ambiguity of the position of the utmost left and utmost right points in the described representation. They might be included both in the upper and lower sub-hull. Both points must belong to the upper sub-hulls of the left and right hulls before finding the tangent line, because otherwise such tangent may be found incorrectly. An example of such an incorrect search is given in Fig. 2.

To avoid such a situation, it is necessary to move the aforementioned points to the upper sub-hulls before merging them. For the rightmost point of the left hull and the leftmost point of the right hull we have the following cases. Similarly to the previous argument, they must be transferred to the upper parts of the hulls. And after merging, these points must be transferred to the lower parts of the hull, if they do not belong to the resulting upper part of the final hull. Otherwise, the formed hull may be incorrect. An example of such a case is shown in Fig. 3.

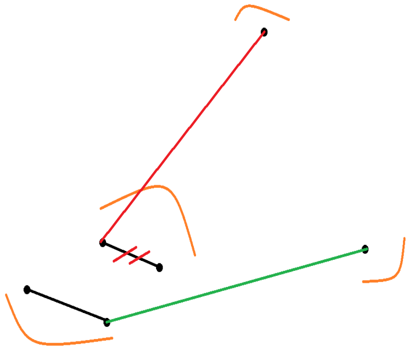


Figure 2: Example of an incorrect position of the utmost left point in the left hull.

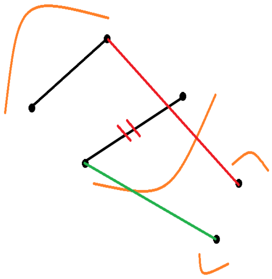


Figure 3: Example of a convex hull with a wrong position of the utmost left point of the left sub-hull.

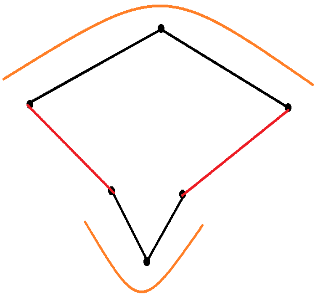


Figure 4: Example of the situation when upper and lower sub-hulls do not form a correct convex hull.

To transfer the utmost left point of the lower sub-hull to the upper sub-hull means to split the concatenable queue representing the lower sub-hull over its utmost left points and merge the obtained part with the upper sub-hull.

After combining the parts of the convex hulls, another corner case might take place. The search for the tangent for the upper parts of the hulls does not take into account the position of the lower parts and vice versa. As a result, the upper and lower parts of the final hull may not form a coherent structure. An example of such a situation is shown in Fig. 4.

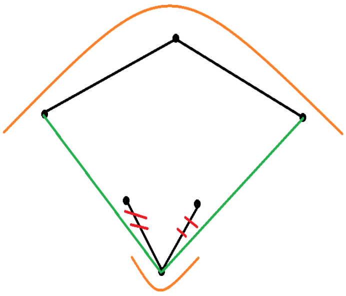


Figure 5: Correctly constructed convex hull. Here the *pivoting left* and *pivoting right* nodes are represented by the lowers points in the convex hull.

To avoid such a situation, it is necessary to perform the step of cutting off the redundant left and right parts of the formed lower sub-hull. To do so, we perform two more binary searches on the lower sub-hull to find *pivoting left* and *pivoting right* nodes. The part of the lower sub-hull in between those pivoting nodes forms the correct hull with the upper sub-hull. Fig. 5 shows example of such procedure.

2.2 “Divide-and-conquer” algorithm interface

The next goal of this work is to build a unified algorithmic environment. The construction of such an object requires the combination of an algorithmic database together with the necessary data structures. It is needed to create an interface for generic algorithms based on the “divide-and-conquer” strategy.

We first list the components of such an interface:

- Preprocessing.
- Dividing task into sub-tasks.
- Merging results of solved sub-tasks.
- Checking if a given input represents a base case.
- Solving the base case.

To construct the final interface it remains to determine the input and output types of its functions. A large number of computational geometry algorithms, such as the minimum spanning tree, the Delaunay triangulation, the Voronoi diagram, and the convex hull accept the list of points. A list can also be easily split into two parts of roughly the same size. The output type of the interface should store result data computed by the algorithmic environment. In our implementation we use a special class *Result* with *convexHull* field. The field stores convex hull computed on a specific step of the recursion.

Listing 1 shows the constructed algorithm model. Here every aforementioned component is represented as a function.

```
interface DaCAlgorithm:
  Points preprocess(Points input)
  Pair[Points, Points] divide(Points input)
  Result merge(Result first, Result second)

  boolean isBaseCase(Points input)
  Result solveBaseCase(Points input)
```

Listing 1: Algorithm model based on the "divide-and-conquer" principle. Here a list of points is denoted as Points

2.3 Sequential and parallel execution

Although this model very accurately describes the class of algorithms, it does not make it possible to solve the problem directly by having input data. This allows us to separate the implementation of the algorithm from how it is executed. Next the principles of sequential and parallel execution are discussed.

When executing sequentially an algorithm, the sub-problems are computed one by one. We first check if the current input is a base case and if so we can directly compute it by calling *solveBaseCase* procedure. Otherwise the input is split with *divide* and the obtained sub-problems are solved sequentially. Finally the obtained results are merged with *merge* procedure.

In parallel execution, we take into account that the individual sub-problems can be calculated independently, which significantly speeds up the execution of the algorithm. To construct the concurrent execution algorithm, we use the following parallel computation abstraction $computeInParallel(function1, function2)$ which runs the functions $function1$ and $function2$ simultaneously. We use it to solve sub-problems obtained after dividing a given input. Other than that parallel version is identical to the sequential one.

From the implementation standpoint, the performance of parallel execution was improved by introducing a limit on the size of sub-tasks that can be calculated in parallel. This allowed us to distribute work between threads more evenly.

3 CONCATENABLE QUEUE IMPLEMENTATION

As shown in [OL81], the concatenable queue is the key data structure for the algorithm described above and is therefore the basis for the UAEM. Now we will focus on how to efficiently implement it for our algorithmic environment.

Concatenable queue is an Abstract Data Type, that supports the following operations:

- insert();
- remove();
- getMinimum();
- contains();
- split();
- concatenate().

By default the elements in a concatenable queue are kept in a certain predefined order [AH74, pp.. 155-157]. In this article the concatenable queue is implemented as a modified balanced binary tree. Its nodes are divided into non-leaf and leaf ones. The leaf nodes contain all points kept in a tree. A *ConcatenableQueue* object maintains pointers to the root of the tree as well as leaf nodes that contain minimum and maximum values in the tree. Every node has $left$ and *right* pointers which point to its left and right child respectively. For the leaf nodes those pointers point to the left and right neighboring leaf nodes or *nil* if the node is utmost in the tree. Additionally every node keeps a pointer $leftMax$ to a node with the largest element in its left sub-tree, which allows us to perform binary search. The node constructor accepts values $leftMax$, $left$ and *right*. The *height* value each node is kept for the balancing during the split and merge operations. Now we will go into details on how this data structure is implemented.

The *contains* operations is pretty straightforward and uses binary search over the tree so its complexity is $O(\log n)$, where n hereafter denotes the numbers of nodes in the queue.

Algorithm of inserting a new element in a queue looks like as follows. First, the position for a new node is searched. Then a new node is inserted between two adjacent leaves. Going back a new non-leaf node is created - the parent for the new node and one of its neighbors. The algorithm is formally described in Listing 2. Here the *updateHeight* subroutine updates the height value for a given node, the $insertLeaf$ insets a new leaf node between two adjacent leaf nodes.

```
Node insert(int v, Node node):
  Node result = nil
  if v > node.leftMax.value:
    if !node.isLeaf:
      node.right = insert(v, node.right)
    else:
      Node newNode = insertLeaf(node, node.right, v)
      result = Node(node, node, newNode)
  else:
        if !node.isLeaf:
            node.left = insert(v, node.left)
        else:
            Node newNode = insertLeaf(node.left,
                  node, v)
            result = Node(newNode, newNode, node)

    if result == nil:
        result = node

    updateHeight(result)

    return result
```

Listing 2: Queue insertion algorithm

```
split(int v, Node node, ConcatenableQueue lq,
      ConcatenableQueue rq):
    if !node.isLeaf
        if v < node.leftMax.value:
            split(v, node.left, lq, rq)
            rq.root = concatenate(rq.root, node.right,
                  node.leftMax)
        else if v > node.leftMax.value:
            split(v, node.right, left, rq)
            lq.root = concatenate(node.left, lq.root,
                  node.leftMax)
        else:
            lq.root = node.left
            lq.maxNode = node.leftMax
            rq.root = node.right
            rq.minNode = node.leftMax.right
            cut(node.leftMax)
    else:
        lq.root = node
        lq.maxNode = node
        rq.minNode = node.right
        cut(node)
```

Listing 3: Queue split algorithm

In the first step we find out if $leftMax$ point to a node with smaller value than v. If so, then, if the current node is not a leaf, the search proceeds on the right sub-tree of the current node. Otherwise, a new leaf is created between the current node and its right neighbor. The case, when $leftMax$ has a greater value than v is analogous. The procedure ends with updating the height on a newly created node, which is returned as a result value.

Since in every step we perform a constant amount of work, the complexity of the procedure is $O(h) = O(\log n)$, where h hereafter denotes the height of the tree. The *remove* operation is analogous.

Now we will discuss the *split* procedure. As an input the procedure takes a value based on which the split is performed, a current node as well as left and right queues, which are constructed as a result of the procedure. The value, by which the split has been performed, belongs to the left queue. The procedure is formally described in Listing 3.

Here the *concatenate* procedure is used. It performs concatenation of two arbitrary nodes and uses their heights to balance the resulting queue. The *cut* procedure breaks connections between two adjacent leaf nodes in a queue and therefore is trivial. In the first step of the *split* operation we check if the current node is not a leaf. Is so, then the procedure continues on either left or right sub-tree. Here a special corner case is considered, where $leftMax$ contains the dividing value. If that is the case, then $leftMax$ becomes the maximum node for the left queue and its right neighboring leaf node becomes the minimum node for the right queue. Finally, if *node* is a leaf, its connections are broken and the value of *maxNode* is updated for the left queue as well as the value of *minNode* for the right queue.

```
Node concatenate(Node ln, Node rn, Node
      leftMax) {
    if ln == nil:
        return rn
    else if rn == nil:
        return ln
    else if ln.height < rn.height:
        rn.left = concatenate(ln, rn.left, leftMax)
        updateHeight(rn)
        return rn
    else if ln.height > rn.height:
        ln.right = concatenate(ln.right, rn, leftMax)
        updateHeight(ln)
        return ln
    else:
        Node result = Node(leftMax, ln, rn)
        updateHeight(result)
        return result
```

Listing 4: Algorithm of merging two queues

The algorithm of the *concatenate* procedure is described in Listing 4. First, we consider corner cases where one of the nodes is *nil*. This is needed to ensure the correctness of the recursion. Then, if the left node is lower than the right, one step down is taken for the right node. If the right node is lower - we take a step down for the left node. If the heights are equal, the joining point is found and a new node must be created. In each step, it is necessary to update the height of the current node because it changes.

We begin analyzing the complexity of the *split* procedure by determining the complexity of the *concatenate* procedure. At each iteration, a step is performed either to the left son of the current node or to the right one. The execution of the recursive procedure finishes by merging two nodes.

Since each step moves us down one level and a constant amount of work is performed for each level, the total complexity of the *concatenate* procedure is $O(h) = O(\log n)$. The *split* procedure uses the *concatenate* procedure as a subroutine. The complexity of a *split* call is equal to the complexity of *concatenate*. The number of recursive *split* calls for one such operation is $O(\log n)$, so the total complexity of the procedure $O(\log^2 n)$. The merge operation of two queues is reduced to the clamping of their root nodes by the *concatenate* procedure, so its complexity is $O(\log n)$.

4 COMPLEXITY ANALYSIS

Theorem 1. *The complexity of the described convex hull construction algorithm for a static set of points is $O(n \log n)$ with sequential execution.*

Proof. We will argue the complexity of the algorithm by listing the complexities of the main steps.

1. Preprocessing: sorting and removal of "inner" repetitions by x and y coordinates $O(n \log n)$.
2. Divide step: splitting list of points in half $O(1)$.
3. Merge step: merging convex hulls obtained from solved sub-tasks $O(\log^2 n)$:
   (a) transfer of the utmost points to upper parts of convex hulls with at most 4 calls to *split* and *merge* operations $O(\log^2 n)$;
   (b) finding the tangent for the upper parts of the hulls with a binary search $O(\log n)$;
   (c) splitting and merging the upper parts with 2 calls to *split* and 1 call to *merge* operations $O(\log^2 n)$;
   (d) moving the utmost points to the bottom of the hulls with at most 2 calls to *split* and *merge* operations $O(\log^2 n)$;
   (e) finding the tangent for the lower parts of the hulls with a binary search $O(\log n)$;
   (f) splitting and merging the lower parts with 2 calls to *split* and 1 call to *merge* operation $O(\log^2 n)$;
   (g) normalization of the obtained lower part with 2 binary searches and 2 calls to *split* operation $O(\log n)$.

Using known algorithms, we can perform sorting in $O(n \log n)$. To estimate the complexity of the recursive procedure for constructing a convex hull, we make the following equation:

$$T(n) = 2T(\frac{n}{2}) + O(\log^2 n) \tag{1}$$

According to result from the theory of algorithmic complexity, we have that the solution of this equation is:

$$T(n) = O(n) \tag{2}$$

Thus, taking into account the preprocessing, we get the total complexity of the algorithm $O(n \log n)$. □

Theorem 2. *The complexity of the recursive convex hull construction is $O(\log^3 n)$ when executed concurrently on $\frac{n}{2}$ processors.*

Proof. The recursion tree has a height of $O(\log n)$ levels. At the lowest level, the number of sub-tasks created is $\frac{n}{2}$. Thus, each sub-task takes no more than $\frac{n}{2}$ time. Next, $O(\log^2 n)$ work is performed at each level. Having the height of the recursion tree, we get the total complexity of the algorithm. □

5 PERFORMANCE EVALUATION

We implement the UAEM with Java programming language using its standard library. We used a machine running Ubuntu 18.04 LTS equipped with 16GB of DDR4/2 RAM and Intel Core i7-8750H CPU which has 6 cores and supports up to 12 threads.

To efficiently solve "divide-and-conquer" subproblems we delegate parallelization to the ForkJoinPool which is available as part of the Oracle's JDK 8. Furthermore, we limit the average number of recursive subproblems per thread. This allows us to better control load balancing for large inputs. We conducted the performance comparison of sequential execution and parallel execution with different numbers of average sub-tasks per thread. Parallel execution was done with 12 thread. All performance data is reported in Table 1.

Parallel computation allows us to achieve up to 37% performance improvement in the best case. Fine-tuning through the average number of sub-problems per thread achieves up to 8% speedup compared to the unbalanced case. Base on collected data we can conclude that the optimal number of subproblems per thread for the convex hull computation is 30.

6 CONCLUSION

We've considered in detail the process of designing and implementing the UAEM as well as the unified data structure for it. In this model a generic interface of a "divide-and-conquer" algorithm was created. This allows to execute the algorithms which are implemented according to this model both sequentially and in parallel. Apart from that a concatenable queue was implemented and served as the basis for the model described above.

Number of points	**Time (μs)**				
	sequential	20 tpth	30 tpth	40 tpth	50 tpth
$1 \cdot 10^5$	62	43	**41**	44	44
$5 \cdot 10^5$	457	309	**288**	311	308
$1 \cdot 10^6$	861	**643**	644	686	672
$5 \cdot 10^6$	3926	2876	**2860**	3005	2973
$1 \cdot 10^7$	6002	5207	**5112**	5465	5120

Table 1: Performance of the convex hull computation in the UAEM. Since our model uses fork-join parallelism we measure how the average number of recursive tasks per thread (tpth) affects the computation time. Bold numbers indicate the best time in each row.

Using the data structure allowed to significantly reduce the time and computational resources for solving the convex hull problem. The main advantages of the developed algorithm are an optimized preprocessing stage and the efficiently implemented merge step, due to the usage of concatenable queue.

The performance comparison for both types of execution shows that the algorithm has a high level of parallelism. We've achieved a speedup of 37% in the best case. It is easy to extend the functionality of the created environment either by adding new or modifying existing algorithms.

7 REFERENCES

[Ima19a] Imagej: An open platform for scientific image analysis. https://imagej.net/Welcome. Accessed: 15.04.2019.

[Ima19b] Imaris software. https://imaris.oxinst.com. Accessed: 15.04.2019.

[ACG*88] Aggarwal, A., Chazelle, B., Guibas, L., O'dunlaing, C. and Yap., C. Parallel computational ge- ometry. Algorithmica, 3(1-4):293-327, November 1988. DOI: 10.1007/BF01762120.

[AH74] Aho, Alfred V. and Hopcroft, John E. The Design and Analysis of Computer Algorithms. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition, 1974.

[AL93] Akl, Selim G. and Lyons, Kelly A. Parallel Computational Geometry. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

[AGR94] Amato, N.M., Goodrich, M.T., and Ramos, E.A. Parallel algorithms for higher-dimensional convex hulls. In Proceedings 35th Annual Symposium on Foundations of Computer Science, pages 683-694, Nov 1994. DOI: 10.1109/S-FCS.1994.365724.

[ACG89] Atallah, M.J., Cole, R. and Goodrich, M.T. Cascading divide-and-conquer: A technique for designing parallel algorithms. SIAM J. Comput., 18(3):499-532, June 1989. DOI: 10.1137/0218035.

[BSV96] Berkman, O., Schieber, B. and Vishkin, U. A fast parallel algorithm for finding the convex hull of a sorted point set. Int. J. Comput. Geometry Appl., 6:231-242, 1996.

[Che95] Chen, D.Z. Efficient geometric algorithms on the erew pram. IEEE Transactions on Parallel and Distributed Systems, 6(1):41-47, Jan 1995. DOI: 10.1109/71.363412.

[CG88] Cole, R. and Goodrich, M.T. Optimal parallel algorithms for polygon and point-set prob- lems. In Proceedings of the Fourth Annual Symposium on Computational Geometry, SCG '88, pages 201-210, New York, NY, USA, 1988. ACM. DOI: 10.1145/73393.73414.

[GJ97] Goodman, Jacob E. and O'Rourke, J., (eds.). Handbook of Discrete and Computational Geometry. CRC Press, Inc., Boca Raton, FL, USA, 1997.

[JaJ97] JaJa, J. An Introduction to Parallel Algorithms. Addison Wesley, 1997.

[OL81] Overmars, Mark H. and Leeuwen, Jan V.. Maintenance of configurations in the plane. Journal of Computer and System Sciences, 23(2):166 - 204, 1981.

[Rei93] Reif, John H. Synthesis of Parallel Algorithms. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1st edition, 1993.

[SSKH11] Sommer, C., Straehle, C., Koethe, U. and Hamprecht, F.A. Ilastik: Interactive learning and segmentation toolkit. In 2011 IEEE International Symposium on Biomedical Imaging: From Nano to Macro, pages 230-233, March 2011. DOI: 10.1109/ISBI.2011.5872394.

[TA10] Tereshchenko V.N. and Anisimov A.V. Recursion and parallel algorithms in geometric modeling problems. Cybernetics and Systems Analysis, 46(2):173-184, 2010.

[Lee90] Leeuwen, Jan V., (eds.). Handbook of Theoretical Computer Science (Vol. A): Algorithms and Complexity. MIT Press, Cambridge, MA, USA, 1990.

Line-Storm Ludic System: An Interactive Augmented Stylus and Writing Pad for Creative Soundscape

Hans Cox
Department of Computer Science
University of Colorado
Colorado Springs, CO
80918, USA
hcox@uccs.edu

Sudhanshu Kumar Semwal
Department of Computer Science
University of Colorado
Colorado Springs, CO
80918, USA
ssemwal@uccs.edu

ABSTRACT

We present *Line-Storm*, an interactive computer system for creative performance. The context we investigated was writing on paper using *Line-Storm*. We used self-report questionnaires as part of research involving human participants, to evaluate *Line-Storm*. *Line-Storm* consisted of a writing stylus and writing pad, augmented with electronics. The writing pad was connected to a contact microphone, and the writing stylus had a small micro-controller board and peripherals attached to it. The signals from these electronic augmentations were fed into the audio-synthesis environment Max/MSP to produce an interactive soundscape. We attempted to discover whether *Line-Storm* enhanced a self-reported sense of being present and engaged during a writing task, and we compared *Line-Storm* to a non-interactive control condition. After performing statistical analysis in SPSS, participants reported they were, on average, no more present and engaged during the experimental condition than during the control condition. As creativity is subtle, and varies with person, time, context, space and so many other factors, this result was somewhat expected by us. A statistically significant result of our study is that some participants responded to *Line-Storm* more positively than others. These Preservers of *Line-Storm* were a group, distinct from other participants, who reported greater presence and engagement and who wrote more words with *Line-Storm* and during the control condition. We discuss the results of our research and place *Line-Storm* in an artistic-technological context, drawing upon writings by Martin Heidegger when considering the nature of *Line-Storm*. Future work includes interesting, immersive, and engaging interactive soundscape for writing or drawing performance, modifying interactive components, improving aesthetics, using more miniaturized electronics, and experimenting with a drawing task instead of a writing task.

Keywords

Ludic System, Creativity, Interactive Soundscape

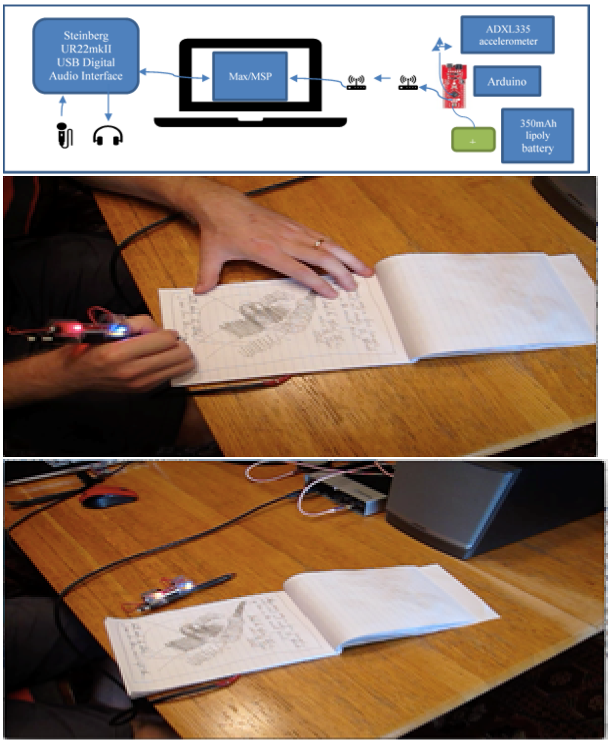


Figure 1: a (above): *Line-Storm* A System Diagram. Fig. 1b-c (below) Line storm stylus drawing and sample of creative content

1 INTRODUCTION

Tod Machover [1-22] [40] has emphasized the need to augment existing, traditional musical instruments while

https://doi.org/10.24132/CSRN.2020.3001.8

ensuring these augmentations act as *stimuli* to the creative process, not simply as additional features. One focus of this paper is to find a way to enhance human creativity. Another is to observe the emergence of the work when the system is used. A third, is our attempt to make something that is fun to use. We have conceived, designed, constructed, evaluated, our system called *Line-Storm*[1], attempting to enhance a sense of both presence and engagement in the user. Only through performance with *Line-Storm*, does *Line-Storm* come into being.

The method of experience sampling–interrupting a person as they go through their daily activities and asking questions about their experience–has been used to find that when people's minds are wandering, they are less happy [37]. "Be Here Now," a mantra popularized in the United States by, for example, Dr. Richard Alpert [16], who became Baba Ram Dass. This mantra now occurs in a leading business publication urging middle managers everywhere to "be present" to be a "great leader" [29] and presumably to reap the rewards of "success." Even the LSD experimentation Dass describes in Be Here Now, carried out on a small, socially acceptable scale in Silicon Valley, where tech workers "microdose" themselves with LSD, to enhance their creativity and improve interpersonal interactions [38]. Some esoteric practices leading to creative work may conjure images of the lone painter or poet, or of a sculptor in her studio. It is not only Silicon Valley technocrats, scrambling for millions and billions of dollars, who might benefit from enhancing human creativity.

> Even now one is ashamed of resting, and prolonged reflection almost gives people a bad conscience. One thinks with a watch in one's hand, while eating meals, and reading the latest news of the stock market; we live today not to miss out on anything. –Nietzsche [45]

Note that Nietzsche was writing well over 100 years before "FOMO," or "fear of missing out," became an expression related to early 21st-century smartphone users. Our point is that we recognize that there are different meanings to the phrase creative work. For example, billionaires and poets are not endorsing the same thing when both use the word "creative" or the word "work," though both may praise "creative work."

Some decry the extreme measures taken by LSD trippers in the 1960s [38], and want to turn the drug into an effective money-making tool. An irony is that creative work translates into fortunes undreamt of by poets such as Robert Frost. There is a story in which Joseph Heller, author of the novel *Catch-22*, when told of an investment banker who had made more money last year than he might ever to be expected to make from the novel, replied that he had something the investment banker would never have: *enough*. So, we argue that it is possible that what was good for Heller, in the anecdote, would probably not have been good for the investment banker, even when the concept of creative work is broadened to include both their endeavors. Enhancing one type of creative work may not enhance the other. The ecstasy of the composer remarked upon by Csikszentmihalyi [14] or of the novelist, may not be found in the same way the "A-ha!" of the software developer is found.

Our work involving *Line-Storm* has been an attempt to provide a ludic system for use by the creative worker. Gaver [21] defines a ludic system as one that is used for its own sake, and not for some other end. By attempting to increase a user's sense of presence and engagement–their being here now–our hope is to provide an immersive environment in which to do creative work with a writing stylus such as the mechanical pencil we chose to use. Taskscape is a complex term from Ingold's "The Temporality of the Landscape" [32], which we will refer to later, when speaking of the new possibilities of a task that *Line-Storm* exposes, as *affordances* in Gibson's sense of the term[19]. One of our committee members, a professor of music, suggested that our work involves the taskscape of the creative worker, working with a writing stylus and paper. This taskscape includes the place, people, and objects surrounding the creative worker doing creative work. The taskscape is social [32]. The experience of the user of our system, and of the research participants who gave of their time to be a part of this paper, is a social experience, and the writing tasks they performed are tasks that fit into "an array of activities"–which include the writing of this sentence [32]. We do not know–as above, because too little work has been done in this area–whether the taskscape of a user of *Line-Storm* is altered in ways more conducive to writing poetry than to the drafting of microprocessor plans, for example, or vice versa. Rather than devise a completely new tool, we have chosen to augment an otherwise ordinary mechanical pencil[2]. Perhaps by looking away from our goal, creative enhancement–as we must when looking at faint night-sky objects with the naked eye [55]–and making the use of the system the primary activity, and the work done with it a secondary activity, we think we will find ourselves progressing in that direction, whereas a direct approach would not have succeeded. By giving a chance for play, we have hoped our system, *Line-Storm*, serves as stim-

[1] We chose the name *Line-Storm* after a favorite Robert Frost poem, "A *Line-Storm* Song."

[2] We could have similarly augmented a paintbrush or a pen, though the paintbrush would have required a different approach. We depend in part on the sounds made by the user's touching of the writing pad, and we cannot expect a paintbrush to make the same level of sound made by a pencil lead.

ulant and facilitator "to the creative process itself," as Machover [40] advises.

2 RELATED WORK

Line-Storm is not digital art as such. Its products are physical objects and phenomena. They are (analog) drawings or writings. It produces sounds, which are–though digitally mediated–analog sounds. The computer is, in *Line-Storm*, an intermediary and a facilitator, with a visual arts component and sound, satisfying criteria for Demers' [8] second sub-genre of sound art. *Line-Storm* amplifies and augments the sonic aspects. The sounds made, while writing or drawing, are captured using a contact microphone and are played through headphones. Sounds of natural phenomena–the sounds of a thunderstorm–augment the writing or drawing experience. These sounds are recorded analog yet are digitally mediated.

2.0.1 Line-Storm as Performance

Line-storm can be used as a form of interactive theater. *Line-Storm* is a piece of evolving art and/or in play [52]. A performer using *Line-Storm* may be using it for different reasons, including for the fun of using it, to write a letter to a friend, to write down a cooking recipe, to write poetry, to draw, because it is a curious thing one wants to understand, or for other reasons. A performance occurs "as action, interaction, and relation" [52]. *Line-Storm* is an interactive system, where the performer's actions cause sounds to occur, which may influence subsequent actions. The sounds can be controlled to some degree, by the performer. The drawing or writing produced during performance is one product of the performance. The sounds, which can be recorded and played back, are another product. The audience of the performance may be the performer alone, or a person or persons witnessing the performance as it occurs, or presented with one or more products of the performance, the written or drawn product or the sound produced. *Line-Storm* is a way of "honoring the ordinary" in Schechner's [52] words.

Play is a way of introducing flow into one's life [53] and has an organic quality. Sounds which we added to *Line-Storm* included those of thunderstorms, which have organic qualities similar to movement of air through a room, or sounds from nearby birds. Thunderstorms followed by quiet rain can be theraputic to some as well. We wanted to use analog (thunderstorm) sounds for our analog ludic system as well. The digital medium is one of permanence and impermanence. One motivation of *Line-Storm* is that it could preserve the practice of the handwritten letter.

Previous work, that investigated augmenting a writing stylus with electronic or computer systems, includes MusicGrip [23], a pressure-sensor-controlled system in which a writing stylus was used to control analog synthesizers. Musc Grip used a one-to-one correspondence between sensor input and synthesizer output. Shichinohe *et al.* [54] used a camera system to implement an augmented-reality system to aid in the instruction of calligraphic writing. Their system monitored brush position and body posture, providing both ambient (color) feedback and verbal feedback. Part of a performance–the *Brain Opera*–the Digital Baton was a wireless baton, augmented with sensors, used as a New Interface for Musical Expression (NIME) [46]. The baton carried an infrared LED at its tip, pressure-sensitive resistors that were controlled by the performer's fingers gripping the baton, and three +/-5g accelerometers. These inputs were mapped to musical parameters. The Digital Baton was a wired NIME, but the authors did discuss what could be done to make it wireless. Tod Machover's work with *hyperinstruments* (electronic augmentations of traditional music instruments) [42]) and [40] are very interesting. Machover's Hyperstring Trilogy [41] was composed and performed using hyperinstruments–hypercello, hyperviolin, and hyperviola–which were traditional classical instruments augmented with sensors. Machover's philosophy of augmenting, and not replacing, traditional tools, is one we have followed in our work [42]. LiveScribe (http://www.livescribe.com), which has produced a wireless pen with handwriting recognition, no longer develops the electronic writing pen it once did, so we did not involve the company's work in our work. Work involving the augmentation of objects other than writing utensils or musical instruments includes the Sonic City system [22], in which the urban environment served as the interface. The Bluetooth Radio Ball Interface (BRBI) [59] augmented a sport ball with sensors, providing sound and music capabilities (mediated by a computer and a Bluetooth radio connection). The Urban Musical Game [48] was another augmentation project involving a sport ball and sound/music generation based on the ball's motions; video of use of Urban Musical Game have been made available on Vimeo (https://vimeo.com/26413625) and (https://vimeo.com/22120867). Measurement of writing motions helps diagnose people suffering from obsessive-compulsive disorder [43]. Handwriting and cell-phone texting have been compared as therapies for Broca's aphasia, with handwriting emerging as the more effective treatment [4]. Embodied cognition models have been used to investigate neural relationships with character writing, copying, and recognition [35]. Preschool children have taken part in fMRI experiments, which demonstrate the importance of "learning-by-doing" approaches to literacy learning, with kinesthetic activity working in tandem with cognition [34]. Existential phenomenology has informed thought regarding the teaching of personal writing, without any technological involvement [33]. Recent work by Kiefer [36] has found neuropsychological evidence for benefits from writing by hand, as opposed to writing using a computer keyboard, including improved learning of reading and writing skills in young children. Morphy and Graham [44] argue students more generally, appear to write better when using word processors than when composing by hand, considering the composition tools (spell check, grammar check, etc.) available in modern word processing software. Al-Ghabra [1] focused on the importance of handwriting for the development of composition skills in college students. Earlier work by Collier and Werier [7] found no difference between high-level characteristics of textual production in proficient adult writers who composed either by hand or while using a word processor.

2.1 Development of Work

Thoreau [58] decried some forms of letter writing, writing that, "The penny-post is, commonly, an institution through which you seriously offer a man that penny for his thoughts which is so often safely offered in jest"; from a different viewpoint, writing letters has been a way for families to stay connected through the generations and has functioned alongside newer media [47]. Twentieth-century German philosopher Martin Heidegger commented, in his Parmenides, upon handwriting, declaring its superiority over use of a typewriter [28]. Philosopher of technology Don Ihde[31] faulted Heidegger for Heidegger's comparison. Philosopher Jacques Derrida [18] also faulted Heidegger, for implying, while emphasizing the importance of "the hand" for humanity, that human beings only have one hand. A typewriter does not offer the affordance [19] of being easily carried up a mountain–although Nietzsche owned a portable typewriter [3] and hiked up the mountains. Likewise, the poem title of a friend, "Notebooks," [39] would read differently if it had to do, not with notebooks, but with some digital note-taking contrivance such as Google's Keep app (http://keep.google.com). Ihde reminds us of the non-transparency of electronic and digital communications media such as the telephone [30], and here, with Nietzsche's typewriter and Gregory Lawless's poem, we see some effects of medium, in practice (typewriter) and in discourse (poem title).

Heidegger [26] decries what he sees as hastiness in the face of a technologically facilitated information glut. Both Heidegger [26] in his "Memorial Address," and Jacques Ellul [20] in *The Technological Society*, declare technology to have become "autonomous" (in Ellul's phrasing), saying its progression could not be stopped, even if human beings *wanted* to stop it. Our thinking here is that technology creates more options, including the option to not use it; non-users of a technology have been considered by Satchell and Dourish [50]. Should we augment human capability, or should we replace it with a technological contrivance? As discussed above, we have followed Machover in choosing to augment human creative capability, using *Line-Storm*.

2.1.1 Creativity and Line-Storm

For work done by Csikszentmihalyi, ninety-one persons were interviewed who were deemed to have made significant contributions to their fields [11]. Many others, who excluded themselves from his study, were skeptical of studying creativity or of participation in the study as being worthy of their time, and some insisted they were too busy being creative to stop and talk about it [13]. A direct approach to enhancing creativity, Csikszentmihalyi [13] writes, is less effective than are attempts to place the creative worker in a favorable environment; but beautiful surroundings are not what he means. The creative worker creates an environment conducive to creative thought and work, despite otherwise unfavorable surroundings; creative people provide a personal pattern" [15]. On the other hand, he denies there is proof that a person needs "delightful" surroundings to engage in creative work [12]. Our work attempts to alter the state of the creative worker, short of accomplishing shamanistic technique or administering psychedelics . Csikszentmihalyi makes a similar claim for the creative worker. The creative worker has their attention focused in areas outside the "status quo" [15]. Creativity is lauded widely yet creativity works for good and bad. Cropley [10, 9] wrote that a computer hacker who circumvents security measures to steal money, has exhibited creativity no less than a symphonic composer imagining a new melodic line. Sternberg [57], writing of what is known about creativity, iterates two points: (1) creativity is mostly "domain-specific," and (2) it is partly independent of measured intelligence quotient (IQ).

2.2 Creativity as Play, and usability

Much ink, including that of Thoreau [58], has been spilled comparing creativity to play. Play does not need the context of a game, to be play. Play may be contrasted with the world of production and work. In attempting to provide an immersive experience conducive to the presence and engagement of the creative worker. Schechner [52] has described the "actual" and the roughness of the performance of writing the poem as "the genuine meeting between performer and problem"; having a sense of presence and engagement is a desirable state for creativity. Dan Ariely [2] wrote that money is a poor motivator to creative production [2]. According to Ariely [2], it is not clear how much of our "mental activity" is under our "direct control," especially when we are working under pressure. Our system might prove more difficult to use than ordinary pencil and paper, and this is not in itself a problem for us because creativity is different than usability. In addition creativity may not be fully mechanizable. Creativity is not only randomness or sheer novelty; it requires filtering by an intelligence [51]. Counterintuitive incentivisation may be called for when attempting to stimulate creativity. Making a task more difficult through the use of unusual tools, may stimulate creative production. Changing the affordances [19] of a once-familiar taskscape [32], may be key to inducing creative thought, making one see a thing or activity in a new way.

2.2.1 Engagement and Preservation in Line-Storm

Line-Storm is interactive. It augments an ordinary pencil and an ordinary pad of paper, adding new interaction possibilities, new affordances [19]. It responds to the person engaged in using it–and it is immersive. The headphones may make it "easy to forget the outside world," allowing the user to "concentrate completely" on the writing task [12] on hand because that is allowable.

Line-Storm is an attempt at providing creative workers with a new tool. Citing Edward Tenner, Runco [49] cautions that tools do not have to be poorly made or poorly designed or have "an undesirable feature, to cause problems" involving either the creative worker or others. Combining technology (pen and soundscape) with art (poem or content) and art works does not, in itself, enhance creativity [49]. In this paper, we argue that for some people Line-Storm provides an opportunity for creativity. Han [24], and Bohme [5], question the place of technologies in our lives, and the role of the associated, technological, perspective in dominating other forms of life. Lucas observed of the world depicted in his 1971 film, *THX 1138*, that "nobody was having any fun, but no one was unhappy" [17]. As mentioned earlier, the enhanced pen/techology provided an opportunity for creativity in Line-Storm. We have made a new piece of technology that is based on fun. We tread softly when we attempt to bring new technologies into the practice of writing or drawing by hand with pencil and paper.

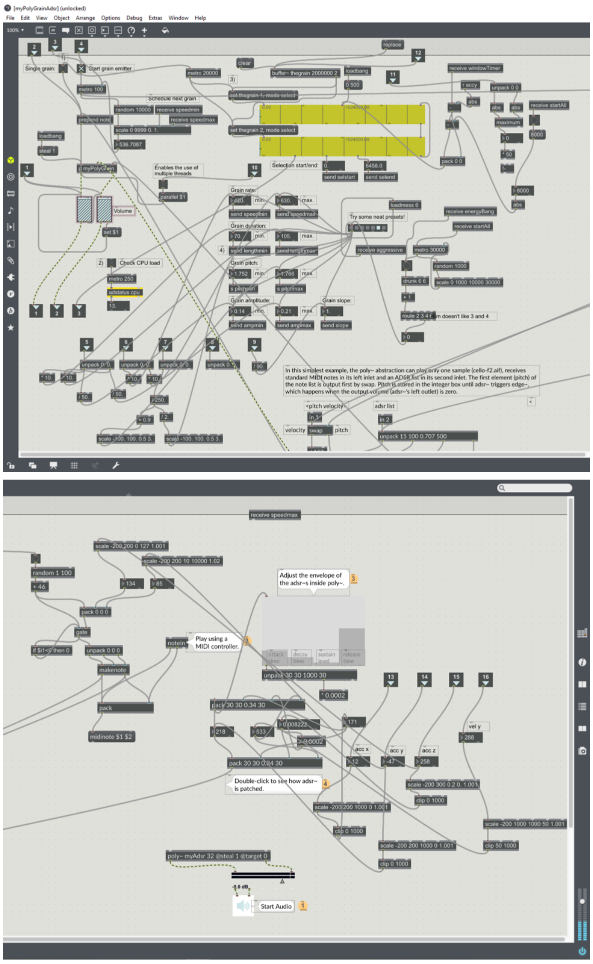
3 IMPLEMENTATION

We present the details of our implementation of *Line-Storm*.

3.1 Development Environment: Max/MSP

We implemented the software interface and sound-synthesis engine of *Line-Storm* using Max/MSP, Version 7.2.3, 64-bit edition. Max/MSP is the mature, commercial successor to Miller's Pure Data (https://puredata.info/downloads/pure-data), a free and open-source project. Like its predecessor, Max/MSP is a graphical programming environment. Objects in the Max/MSP GUI windows can be interconnected and otherwise manipulated inside patchers (graphical representations of program files in Max/MSP) (Figure 2).

Max/MSP has further advantages over some other music-synthesis DAWs such as FM8; Max/MSP is programmable, and it is well-documented.



In this simplest example, the poly~ abstraction can play only one sample (cello-f2.aif), receives standard MIDI notes in its left inlet and an ADSR list in its second inlet. The first element (pitch) of the note list is output first by swap. Pitch is stored in the integer box until adsr~ triggers edge~, which happens when the output volume (adsr~'s left outlet) is zero.

Figure 2: above): *Line-Storm* (above) Granular synthesis engine in Max/MSP (ADSR envelope generator offscreen to right); (below)ADSR envelope generator for granular-synthesis engine.

3.2 Sensor-Fob Construction

The sensor-fob, shown above, in Figure 1(a-c), comprises multiple PCB circuit-boards, powered by a lithium-polymer battery, and five solid-core, insulated copper wires soldered between two of the PCB circuit-boards.

The primary board is an Arduino Fio v3 microcontroller board. This type of Arduino board includes a socket into which an XBee radio transceiver module can be inserted. The Fio v3 can control the inserted XBee radio transceiver module. We have a Digi International XBee radio transceiver module (type S1) inserted into the socket of our Fio v3. The Fio v3 has multiple GPIO/ADC (general purpose input/output or analog-to-digital converter) pins, three of which we have soldered wires to. These three wires are soldered at their ends, to an Adafruit ADXL335 3-axis accelerometer, to its three, analog output-signal pins. Two more pins, and two more wires, connect Vcc and GND on the Fio v3 and ADXL335. The wires are rigid; they both connect the boards and hold them in constant positions relative to each other, in a fixed orientation, by soldering connections between the Fio v3 and the ADXL335.

3.2.1 Accelerometer

We used an Adafruit ADXL335 3-axis accelerometer mounted to a breakout board. Moving the sensor-fob, with its attached ADXL335 unit, in as violent a manner as we were able to do while holding it with a hand, we sometimes reached minimum and maximum sensor-output values, but not always; reaching these values was difficult. Lesser motions were well within the +/-3g range, giving sensor output values below the approximately 1000 maximum and above the approximately 0 (zero) minimum. Sensor output values, raw from the GPIO/ADC pins, range from 0 to approximately 1000, with a center value of approximately 500. This range is compatible with an 8-bit ADC, which the Fio v3 uses. Values below about 500 indicate negative accelerations relative to the corresponding sensor axis, while values above 500 indicate positive accelerations relative to the corresponding sensor axis.

4 EVALUATION: IRB STUDY

Our study involved participation by thirteen persons, but data for one of these participants was discarded, leaving twelve participants with valid data. We had roughly half female and half male, including one who chose not to self-identify. Participant ages are ranged from 18 years to 34 years.

5 DATA ANALYSIS AND RESULTS

As creativity and engagement is somewhat fleeing and vary from one person to another, research hypothesis, that participants' sense of presence and engagement would be greater during the experimental, interactive condition than during the control, non-interactive condition was not directly supported, instead it was only supported by a smaller group of our participants. This was not so unexpected result.

We performed Pearson correlations, and found several statistically significant correlations, discussed below. For example, those participants who reported they lost track of time during the experimental condition also tended to write more during the experimental condition. There was a non-significant correlation between losing track of time and word count during the control condition.

To perform our statistical analyses, we used IBM's SPSS Statistics, Version 25 (https://www.ibm.com/products/spss-statistics), because it is an industry standard statistics processing application.

5.1 Summary of statistical analysis

- There were strong, significant ($p<0.01$) correlations between the initial, baseline level of a sense of presence and engagement and response items 4 (NAT) and 7 (ADJEXP), for both control and experimental conditions.

- A sense of presence and engagement correlated strongly and significantly ($p<0.01$) with adjustment to the "control devices" (augmented stylus, augmented writing pad) (ADJCTL) for both control and experimental conditions.

- There were strong, significant correlations between a sense of the naturalness of interactions with the system and baseline sense of presence and engagement, ease of adjustment to the system experience, and ease of adjustment to the control devices (ADJCTL), for both control and experimental conditions. We found that participants found their interactions with the system more natural during the control condition, and less natural during the experimental condition. Participants adjusted to the system experience more quickly during the control condition than they did during the experimental condition.

- There was a group of participants who responded more favorably to the experimental condition than the rest of the participants (analysis performed using K-means clustering tests). *This is a significant result for our experiments.*

- Those who wrote more in the control condition wrote more in the experimental condition. *This is also a significant result for our experiments.*

- The more participants lost track of time, in the experimental condition, the more they wrote–or vice versa. *This is a significant result for our experiments.*

- We found correlations between a sense of presence and engagement during the experimental condition (PANDE1), and the degree to which a participant lost track of time while using the system during the experimental condition. *This is a significant result for our experiment, and we call these participants as the* **Preservers of Line-Storm** .

Our findings indicate that there appears to have been a significant group of participants, roughly half the participants, the Preservers of *Line-Storm*, who became immersed during the experimental condition. These participants tended to write more during the control and experimental conditions, they tended to experience the sound components of the system (control and experimental) in a way that led to their reporting less prominence of the visual aspects of the system, and they tended to lose track of time during the experimental condition.

It seems likely that attention fluctuates over time, and the mind naturally wanders and returns. Future work would include investigation of the ways such natural fluctuations in attention would be relevant to our work. Considering the ordering of questionnaire completion was nearly always the same (Demographic, Experimental, Control), natural fluctuations in attention (and presence and engagement) may help to explain our results. Considering how we might have been wearing out our participants, by making demands upon their attentional resources, future work might be done that minimized attentional fatigue.

First, as equipment, *Line-Storm* is a tool we have made for a purpose. As equipment, *Line-Storm* has a thingly character and an equipmental character. As a thing, *Line-Storm* exists as an object that can be encountered in the world, like a rock.

As art work, the art-work is the creation *Line-Storm* due to interaction – whatever the user does is recorded on the pad and the performance is the sound created for the user so that they are engaged perhaps because of the sound. As art work, *Line-Storm* is bringing forth of the work that there lies this offering that it be [27]. When we evaluated *Line-Storm* in terms of its capacity for leading to a possible increase in self-reported presence and engagement, we treated it as equipment. Yet some participants, while using *Line-Storm*, treated it not as equipment but as art work. Hence, we will refer to the group of participants who gave higher ratings to *Line-Storm*, and who wrote more while using it, as the Preservers of *Line-Storm*. The Preservers let *Line-Storm* be what it is. Without the Preservers, *Line-Storm* "cannot itself come into being" [27]. We use Heidegger's nation of "preserver," someone who allows the work to be what it is, who brings themselves to the work, Line-Storm, letting themselves experience the writing task and familiar materials?pencil and paper?as if for the first time ([27].

5.2 Performance Affordances in *Line-Storm*

Line-Storm permits itself to be used in performance. A performance with *Line-Storm* could be understood to point out the overlapping of sensory or perceptual modes commonly thought of as separate. Seeing, hearing, moving, and proprioception involve cross-modal transfer [56]. The sound and visual aspects overlap more strongly in *Line-Storm* than in ordinary writing or drawing, because of the amplification of what had been quiet sounds, i.e. the sound made by stylus on the paper which was amplified and merged with other sounds, such as thunderstorm. *Line-Storm* makes affordances prominent, in the writing stylus and writing pad, that may not have been apparent: their sound-producing capabilities, which can be used in a performance. Preservers of *Line-Storm* find its affordances.

5.3 Robotany and *Line-Storm*

In 2006, a living Japanese maple tree was augmented with nitinol wires and optical and audio sensors. The tree moved its branches, using the nitinol "muscle" wires, in response to the presence of people detected by its sensors. When the mechanical components were hidden from view, during the first exhibit, interactions took place with people treating the tree as ready-to-hand. The mechanical components of Breeze withdrew, became transparent, and the people at the first exhibit interacted with Breeze as an interactive art work. During the second exhibit, with the mechanical components poorly hidden by the tall, open shape of the mountain laurel, attendees at the exhibit tended to comment on the engineering of Breeze instead of interacting with it freely as had the attendees during the first exhibit. This is quite interesting for *our* implementation as well.

6 A LUDIC ENVIRONMENT FOR THE PRESERVERS OF LINE-STORM

We conceived our work, initially, as an entertainment system, to be used for one's own pleasure while writing in a journal. We followed that by hoping to jolt users out of complacent acquaintance with paper and pencil and present the writing tools and writing situation as if for the first time, to encourage the practice of writing and sending handwritten letters. We finished the work by attempting to enhance human creativity when working with a writing stylus and paper writing pad, by increasing participants' sense of presence and engagement. We found correlations and K-means clustering results that did suggest there was a group of participants who responded favorably to *Line-Storm*.

We expected that a direct approach to enhancing creativity may/would fail; we attempted to construct a system the use of which would be an end and not only a means [21], and hoped this might lead, indirectly, to enhancing creativity by encouraging play and playfulness. We provided a ludic environment for creative work, in which some users would focus on using the system, not expecting an outcome and will create their own play/outcome and accept what emerges or not–no quest, no winners, no points or gold to deliver outcome-based satisfaction. In a ludic system, therefore, the creative work (outcome is what it is) and the results would be a secondary consideration and may emerge by itself, an indirect result of the use of the system. We hoped participants in our experiments would find themselves "losing themselves," and a group of participants did tend to lose track of time while they used or performed with *Line-Storm*. We believe these participants became more absorbed while using the experimental system, exactly our intention. Losing oneself while using the system might open one up to creative energies, thoughts, feelings, and actions that would ordinarily not occur, as Nietzsche [45] wrote.

The Preservers of Line-Storm differ from the other experimental participants. They have a type of knowing (pg. 192 [27]). They allow themselves to explore the world opened by Line-Storm (pg. 169 [27]). They "[let] the work be a work" (pg. 191 [27]). They heed the call of Line-Storm, noticing its statement: "that it is" (pg. 190 [27]). For them, pencil and paper, and the writing task in which they engage, is new, fresh, and they experience what had become a commonplace activity, anew (pg. 171 [27]).

7 CONCLUSIONS AND FUTURE WORK

One component of *Line-Storm* is its interactivity. A certain type of actions and intentions are required on the part of the participant for *Line-Storm* to be what it is. Coffin wrote of Breeze, and of interactive systems more generally, that interactions with them may be "effortless, unscripted, emergent, and engaged" if the mapping of responses is well done with respect to our "meaning-making sensibilities" [6]. Our goal that *Line-Storm* would provide for increased presence and engagement was not met for all participants. Still, some participants appeared to have had fun and play while using *Line-Storm*. Some of these participants likely experienced *Line-Storm* as art work, and so we would have found *preservers* for our work, who brought out its workly character, and who would belong to it just as we belong to it as its creators. This justifies our efforts. The participants' prior knowledge is relevant when considering their responses to *Line-Storm*. *Line-Storm*, as a tool, exists not by itself but among a constellation of related tools; those related tools, some of which a given participant may be familiar with, and some of which they may not be familiar with, allow *Line-Storm* "to be this equipment that it is" [25]. A participant's degree of familiarity with related tools, such as an envelope, stamp, mail-box, and pencil and paper, help to determine what *Line-Storm* is for that participant. We see a nearly significant (r = 0.620, p = 0.056) correlation of current writing or drawing (by hand) practice to number of words written while using *Line-Storm*. We think that, this correlation indicates participants who regularly wrote or drew by hand were better able to experience *Line-Storm* as it was intended, and see its' authenticity.

Future work would include the following items, listed as follows: (a) We would reinstate the capability of triggering multiple thunderstorm samples in rapid sequence. (b) Make a cover for the electronic components on the sensor-fob. (c) Extension to a mobile platform (d) Investigate the use of more miniaturized RF components. We do not need the relatively large antennae of the XBee radios, which can operate over a larger distance than we envision for the use of our system. Bluetooth would provide the necessary range. (e) Investigate using more miniaturized micro-controller boards. The Arduino Fio v3 was the smallest board we found, when we began our work, with all the functionality we needed. A smaller board would make a less intrusive sensor-fob. (f) Experiment with different styli, including a paintbrush, a child's crayon, a marker, a piece of chalk, a paint roller, and so on. Attaching a contact microphone to the surfaces used with many of these would probably produce a suitable-strength vibration for use with our system. (g) Experiment with a baton-type stylus like the one used by Paradiso and Machover in the Brain Opera. (h) Investigate a wrist-worn appliance to augment or replace the motion-tracking capability of the stylus sensor-fob. (i) Gather more data involving a larger sample size. (j) Vary the type of music listened to during the control condition. (k) Consider ways to run experiments without wearing out participants by making excessive demands on their attention. (l) Experiment with a multi-user system. Users could be situated in the same place or could communicate via a computer network such as the internet. (m) Collaborate with music composer of electroacoustic music. Collaborating with a person skilled in the creation of electronic music would be of great benefit in future as well.

Finally, it has occurred to us that Line-storm as an *augmentation itself is innovative*. Augmenting means a possibility that is completely different than the original which Line-Storm is. The *Preservers of Line-Storm*, in our experiments, showed that there is promise for our augmented interface. Creativity is difficult to capture and define. Still, our work provided a completely different experience through augmented interaction to creative writing which enhanced the user experience, enhancing ordinary pen and paper.

8 REFERENCES

[1] I. M. Al-Ghabra. Handwriting: A matter of affairs. *English Language Teaching*, 8(10):168–178, 2015.

[2] D. Ariely. *The Upside of Irrationality*. HarperCollins, New York, 2010.

[3] S. Avnskog. Friedrich nietzsche and his typewriter - a malling-hansen writing ball), February 2008.

[4] P. M. Beeson, K. Higginson, and K. . Rising. Writing treatment for aphasia: a texting approach). *Journal of Speech, Language, and Hearing Research*, 56(3):945, 2013.

[5] G. Böhme. *Invasive Technification: Critical Essays in the Philosophy of Technology*. Bloomsbury, New York, 2012.

[6] J. Coffin. Robotany and lichtung: a contribution to phenomenological dialogue. In *Proceedings of the Second International Conference on Tangible and Embedded Interaction*, pages 217–220, Bonn, Germany, 2008. ACM.

[7] R. Collier and C. Werier. When computer writers compose by hand. *Computers and Composition*, 12:47–59, 1995.

[8] N. Collins, M. Schedel, and S. Wilson. *Electronic Music*. Cambridge UP, Cambridge, England, 2013.

[9] A. J. Cropley. Acknowledgements. In *The Dark Side of Creativity*, pages 1–14. Cambridge UP, Cambridge, UK, 2010.

[10] A. J. Cropley. The dark side of creativity: What is it? In D. H. Cropley, editor, *The Dark Side of Creativity*. Cambridge UP, A. J, 2010.

[11] M. Csikszentmihalyi. Acknowledgements. In *Creativity: Flow and the Psychology of Discovery and Invention*, pages vii–viii. HarperCollins, New York, 1996.

[12] M. Csikszentmihalyi. Creative surroundings. In *Creativity: Flow and the Psychology of Discovery and Invention*, pages 127–147. HarperCollins, 1996.

[13] M. Csikszentmihalyi. Setting the stage. In *Creativity: Flow and the Psychology of Discovery and Invention*, pages 1–20. HarperCollins, 1996.

[14] M. Csikszentmihalyi. Flow, the secret to happiness, 2004.

[15] M. Csikszentmihalyi. Motivation and creativity: Towards a synthesis of structural and energistic approaches to cognition. In *Flow and the Foundations of Positive Psychology: The Collected Works of Mihaly Csikszentmihalyi*, pages 155–173. Springer, New York, 2014.

[16] R. Dass. *Be Here Now*. Lama Foundation, San Cristobal, NM, 1971.

[17] M. T. Decker. They want unfreedom and one-dimensional thought? i'll give them unfreedom and one-dimensional thought: George lucas, thx-1138, and the persistence of marcusian social critique in american graffiti and the star wars films. *Extrapolation*, 50(3):417–441, September 2009.

[18] J. Derrida. Geschlecht ii: Heidegger's hand. In J. Derrida and J. Sallis, editors, *Deconstruction and Philosophy: The Texts of Jacques Derrida*, pages 161–196. Univ. Chicago Press, Chicago, 1987.

[19] P. Dourish. Being-in-the-world. In *P. Dourish, Where the Action Is: The Foundations of Embodied Interaction*, pages 99–126. MIT Press, Cambridge, MA, 2001.

[20] J. Ellul. *The Technological Society*. Alfred A. Knopf, 1964.

[21] B. Gaver. Designing for homo ludens, still. In T. L. J. Binder, editor, *(Re)Searching the Digital Bauhaus*, pages 163–178. Springer-Verlag, London, 2009.

[22] L. Gaye, R. MazÃ©, and L. E. Holmquist. Sonic city: The urban environment as a musical interface. In *Proceedings of the 2003 Conference on New Interfaces for Musical Expression (NIME-03)*, pages 109–115, Montreal, Quebec, Canada, 2003. ACM.

[23] N.-w. Gong, M. Laibowitz, and J. A. Paradiso. Musicgrip: A writing instrument for music control. In *Proceedings of the International Conference on New Interfaces for Musical Expression 2009*, pages 74–77, Pittsburgh, PA, 2009. New Interfaces for Musical Expression.

[24] B.-C. Han. *Psychopolitics: Neoliberalism and New Technologies of Power*. Verso, 2017.

[25] M. Heidegger. *Being and Time*. Harper Perennial, New York, NY, 1962.

[26] M. Heidegger. Memorial address. In *Discourse on Thinking*, pages 43–57. Harper & Row, 1966.

[27] M. Heidegger. The origin of the work of art. In D. F. Krell, editor, *Martin Heidegger: Basic Writings*, pages 139–212. HarperSanFrancisco, San Francisco, 1993.

[28] M. Heidegger. Concealment and forgetting. In A. Schuwer and R. Rojcewicz, editors, *Parmenides*, pages 77–86. Indiana UP, 1998.

[29] R. Hougaard and J. Carter. If you aspire to be a great leader, be present. *Harvard Business Review*, 29, June 2017.

[30] D. Ihde. The technological embodiment of media. In *Existential Technics*, pages 47–63. SUNY

Press, 1983.

[31] D. Ihde. Postphenomenological re-embodiment. *Found Sci*, 17:373–377, 2012.

[32] T. Ingold. On the temporality of the landscape. *World Archaeology*, 25(2):152–174, October 1993.

[33] L. Jacobs. Existential phenomenology and personal writing. *College Composition and Communication*, 26(3):293–297, October 1975.

[34] K. H. James. Sensori-motor experience leads to changes in visual processing in the developing brain. *Dev Sci*, 13:2, March 2010.

[35] K. H. James and I. Gauthier. Letter processing automatically recruits a sensory-motor brain network. *Neuropsychologia*, 44:2937–2949, 2006.

[36] M. Kiefer, S. Schuler, C. Mayer, N. M. Trumpp, K. Hille, and S. Sachse. Handwriting or typewriting? the influence of pen- or keyboard-based writing training on reading and writing performance in preschool children. *Advances in Cognitive Psychology*, 11(4):136–146, 2015.

[37] M. A. Killingsworth and D. T. Gilbert. A wandering mind is an unhappy mind. *Science*, 330(6006):932, November 2010.

[38] H. Kuchler. How silicon valley rediscovered lsd. *Financial Times*, 29, August 2017.

[39] G. Lawless. Notebooks. In *Foreclosure*, page 10. Back Page Books, Waltham, MA, 2013.

[40] T. Machover. *Hyperinstruments: A Progress Report, 1987-1991*. MIT Media Laboratory, Cambridge, MA, 1992.

[41] T. Machover. *Hyperstring Trilogy [Recorded by A. Kavafian, K. Kashkashian, & M. Haimovitz]*. Oxingale Records, 2004.

[42] T. Machover. Dreaming a new music. *Chamber Music*, 23(5):46–54, 2006.

[43] P. Mavrogiorgou, R. Mergl, P. Tigges, J. El Husseini, A. Schroter, G. Juckel, and M. Zaudig. Kinematic analysis of handwriting movements in patients with obsessive-compulsive disorder. *J Neurol Neurosurg Psychiatry*, 70:605–612, 2001.

[44] P. Morphy and S. Graham. Word processing programs and weaker writers/readers: a meta-analysis of research findings. *Reading and Writing*, 25(3):641–678, 2012.

[45] F. W. Nietzsche. *The Gay Science: With a Prelude in Rhymes and an Appendix of Songs*. Vintage Books, New York, NY, 1974.

[46] J. A. Paradiso. The brain opera technology: New instruments and gestural sensors for musical interaction and performance. *Journal of New Music Research*, 28(2):130–149, 1999.

[47] E. Pecorale. *Writing to Maintain Relationships: From Letter Writing to Facebook: Crossing Borders, Generations, and Mediums*. PhD thesis, ProQuest Dissertations Publishing, 2012.

[48] N. Rasamimanana, E. FlÃ©ty, F. Bevilacqua, A. Cera, J. Bloit, and U. Frechin, J.-L. ... Petrevski. The urban musical game: Using sport balls as musical interfaces. In *CHI*, pages 1027–1030, Austin, TX, 2012. ACM.

[49] M. A. Runco. Enhancement and the fulfillment of potential. In M. A. Runco, editor, *Creativity: Research, Development, and Practice*, pages 319–373. Academic Press, 2007.

[50] C. Satchell and P. Dourish. Beyond the user: Use and non-use in hci. In *OZCHI 2009: Design: Open 24/7: 21st Annual Conference of the Australian Computer-Human Interaction Special Interest Group (CHISIG) of the Human Factors and Ergonomics Society of Australia (HFESA)*, Melbourne, 2009. The University of Melbourne.

[51] K. R. Sawyer. The emergence of creativity. *Philosophical Psychology*, 12(4):447–469, 1999.

[52] R. Schechner. Actuals. In R. Schechner, editor, *Performance Theory*, pages 26–65. Routledge Classics, New York, NY, 2003.

[53] R. Schechner. *Performance Studies: An Introduction*. Routledge, New York, 2006.

[54] T. Shichinohe, T. Yamable, T. Iwata, and T. Nakjima. Augmented calligraphy: Experimental feedback design for writing skill development. In *TEI'11*, pages 301–302, Funchal, Portugal, 2011. ACM.

[55] C. Springob. Why is it easier to see a star if you look slightly to the side? *Ask an Astronomer*, Jun 2015.

[56] N. Stern. *Interactive Art and Embodiment*. Gylphi Ltd, Canterbury, England, 2013.

[57] R. J. Sternberg. Preface. In R. J. Sternberg, editor, *Wisdom, Intelligence and Creativity Synthesized*, pages ix–xviii. Cambridge Univ. Press, New York, NY, 2003.

[58] H. D. Thoreau. Where I lived, and what I lived for. In H. D. Thoreau, editor, *Walden, or Life in the Woods*, pages 67–79. Collier Books, New York, 1962.

[59] W. S. Yeo. The bluetooth radio ball interface (brbi): A wireless interface for music/sound control and motion sonification. *International Computer Music Conference 2006 (ICMC 2006)*, 2006.

Modeling of ultrasound tomographic imaging for non-destructive inspection of underwater structures

Igor Belykh
Peter the Great St. Petersburg Polytechnic University
Polytechnicheskaya, 29
Russia, 195251, St. Petersburg
Igor.Belyh@cit.icc.spbstu.ru

Ivan Zakharov
Peter the Great St. Petersburg Polytechnic University
Polytechnicheskaya, 29
Russia, 195251, St. Petersburg
zakharoiv@gmail.com

Egor Lebedev
Peter the Great St. Petersburg Polytechnic University
Polytechnicheskaya, 29
Russia, 195251, St. Petersburg
mattspring0@gmail.com

ABSTRACT

In non-destructive inspection regular ultrasound techniques provide detection and estimate size of specific defects or undesired objects in different structures. The method of tomographic imaging modeled in this paper can provide the correct position and shape of the object inside underwater structures to be inspected. The proposed method is based on 2-D full waveform inversion of reflected ultrasound field registered by the array of sensors at a short distance from the target object. The computationally intensive algorithm for tomographic image reconstruction is described and tested with focus on underwater pipeline ultrasound inspection. The results of numerical modeling using parallel calculations and physical modeling are presented and discussed.

Keywords
Ultrasound tomography, waveform inversion, physical modeling, non-destructive inspection.

1. INTRODUCTION

The aim of ultrasonic (US) non-destructive testing and inspection (NDI) methods is to detect the presence or absence of flaws in the form of internal defects or objects in materials, infrastructure and equipment in various industries [Bli96]. Among the NDI methods aimed at studying defects are echo-pulse (reflected waves), shadow (transmitted waves) and diffraction (diffracted waves). The most reliable method of ultrasound phased arrays uses lines or matrices of piezoelectric transducers to produce focused US beams for flaw detection in NDI and medicine. These methods can determine the position and estimate size of the target object, but not a detailed information about its shape, orientation and configuration. In essence, US NDI methods are aimed at revealing the structure of inhomogeneous medium from the registered wave field. However, this inverse problem solution often come down to comparing the recorded wave patterns with the referenced ones for the most frequently encountered defects, which does not always lead to satisfactory results [Zie18].

The use of ultrasound tomography (UST) has good potential for the NDI in terms of determining the detailed configuration and physical properties of the specific defects without a priori information about them.

When elastic waves propagate in heterogeneous media, various effects distort the wave field - these are multiple reflections of waves from internal boundaries, wave front migration due to refraction, energy scattering and absorption in various materials, and also the effect of wave diffraction on inhomogeneities. Elimination of diffraction artifacts allows to more accurately determine the shape and position of objects and identify small objects in ultrasound images [Bel17]. In US phased arrays diffraction effects are minimized at physical level. Implementations of theoretical approaches to suppressing reverberations and diffraction distortions were developed for seismic data processing problems solution. These methods were popular when the computing power was not enough for a full numerical solution of the wave equation [Vir09].

2. RECENT SOLUTIONS

Despite current approaches to solving inverse problems by the waveform inversion method in medical applications [Li08] and NDI [Sei17] being computationally complex, they are very promising because they minimize all distorting effects, allowing to obtain correct spatial image of the internal structure of the object under study, including mapping of absorption coefficient.

Over the past decade a significant theoretical and practical progress has been achieved in solving the problems of UST for medical applications. Most

https://doi.org/10.24132/CSRN.2020.3001.9

solutions employ a toroidal transducer system to register total wave field of both refracted and reflected waves. The inspected medium has to be inside the tor, but this imposes restrictions on its size. The approaches to UST differ not only in types of the waves used, but also in methods of reconstructing spatial images from a recorded wave field. There were efforts to solve the wave equation by difference schemes in time domain [Gon17] or wave inversion in frequency domain [San15]. The above approaches of obtaining tomographic images are very computationally intensive, but are close to practical implementation [Hua16] because in recent years, massive parallel computing systems such as GPU clusters and supercomputers have been rapidly developing.

In this paper a method for obtaining an image of 2-D inhomogeneous medium is modeled by registering only reflected wave field at a certain distance from the object of interest with an array of transducers and its subsequent inversion. This acquisition scheme does not impose restrictions on the size of the object and allows to build a high-resolution volumetric ultrasound image with a set of two-dimensional tomographic slices. The paper presents the results of solving forward and inverse problems of ultrasound image reconstruction based on 2-D inversion of reflected wave field in time domain and explores the applicability of this method in non-destructive inspection technology. It is known that time-reversal reconstruction has a slightly improved signal to noise ratio [Tre14] compare to frequency-reversal one. Numerical modeling is supported by physical laboratory experiments.

3. PROPOSED SOLUTION

Forward Problem

The forward problem is formulated for the heterogeneous elastic half-space with an array of N transducers on its surface and a given shape of the pulse emitted by the source. Each transducer s alternately emits a given pulse, and all N transducers, including s, receive reflected waves from the medium during the recording time T.

The forward problem consists in finding a numerical solution of the wave equation in the acoustic approximation, having the form:

$$\frac{1}{c^2(x)}\frac{\partial^2 p(x,t;x_s)}{\partial t^2} - \nabla^2 p(x,t;x_s) = s(x,t;x_s), \qquad (1)$$

where x_s is the position of the pulse source, $p(x,t;x_s)$ is the pressure wave amplitude at point x at time t; $c(x)$ is the speed of sound at the point x, $s(x,t;x_s)$ is the initial signal.

There are various approaches to the numerical solution of such an equation, for example, the classical finite difference method mentioned above. Although, pseudo-spectral method [Ver18], in which spatial derivatives are approximated by the Fourier transform, is more stable and less resource-intensive. The solution of forward problem with N positions of the source is a wave field in the form of a set of N frames, i.e. ensembles of data received from one source, and each frame consists of N records, calculated for every receiver.

Inverse Problem

The input data for the inverse problem are wave field records obtained as a result of forward problem solution or physical experiment. They are stored in the form of N multichannel frames of finite time T. The solution of the inverse problem is spatial distribution of the acoustic parameter of the inhomogeneous medium restored from the original reflection wave field. In this case, the acoustic parameter is proportional to reflection coefficient at each point of two-dimensional space.

The main idea of the waveform inversion method is to use knowledge of the mathematical model of elastic wave propagation for numerical modeling of the physical process and finding those parameters of the equation that allow the most accurate simulation of data obtained as a result of a numerical or physical experiment.

The wave inversion problem is formulated as the problem of minimizing some objective function characterizing the deviation δp of the simulated data from the experimental data. The objective function was taken as [Liu17]:

$$E(c) = \frac{1}{2}\sum_s \sum_r \int_0^T [\delta p(x_r,t;x_s)]^2 dt, \qquad (2)$$

$$\delta p(x_r,t;x_s) = p(x_r,t;x_s) - p_{obs}(x_r,t;x_s), \qquad (3)$$

where s is the number of the signal source; r - signal receiver number; c is the current velocity model; $p(x_r,t;x_s)$ and $p_{obs}(x_r,t;x_s)$ are modeled and observed data received by receiver r at time t when emitted by source s.

To determine the direction of minimization at each iteration, the gradient of the function E is calculated in the approximation $g^{(k)} \approx \nabla E(c^{(k)})$, after which the current velocity model is shifted by the following formula:

$$c^{(k+1)} = c^{(k)} + \alpha^{(k)} d^{(k)}, \qquad (4)$$

where $c^{(k)}$ is the velocity model built at the k-th iteration; $\alpha^{(k)}$ is the minimization step; $d^{(k)}$ is the direction of minimization, which depends on the optimization strategy, and in the used conjugate gradients (CG) method is defined as:

$$d^{(k)} = -g^{(k)} + \beta^{(k)} d^{(k-1)}, \quad (5)$$

where $\beta^{(k)}$ is a parameter that is selected depending on the implementation. The CG method refers to locally optimizing, because depending on the choice of the initial approximation of the velocity model $c^{(0)}$, the algorithm can converge to an extremum that is not a global minimum of the objective function. This is especially evident when solving nonlinear problems, which include the problem of data inversion using the wave equation. The execution of each iteration of the algorithm involves the calculation of two main parameters: the gradient vector and the scalar value of the optimization step. CG method is efficient but its disadvantage is that it needs to store not only sensor data, but also the total wave field of the solutions of the main and conjugate equations that significantly increases the amount of data stored in memory.

One of the ways to overcome the instability of optimization problems is regularization, i.e. adding some additional term to the error function condition. Most often, regularization is a function that depends on some a priori data about the model, in this case, the initial distribution of velocities. For waveform data inversion, it is important that the regularizing function preserves the edges of objects in the image for greater resolution. Therefore, the total variation (TV) regularization was used. The regularized error function has the following general form:

$$E_R(c) = E(c) + \lambda R(c), \quad (6)$$

where λ is the regularization coefficient, which determines the degree of influence of the regularizing function on the result. The coefficient value is determined empirically for a specific task. TV-regularization is often used for tomographic images reconstruction and its function has the following form [Lin12]:

$$R_{TV}(c(x)) = \sum_x \sqrt{\varepsilon^2 + ||\nabla c(x)||^2}, \quad (7)$$

where x is the spatial coordinate, $c(x)$ is the velocity distribution, ε is small a term providing differentiability with the norm equal to zero in the entire region. In our case it was chosen as 0.01, lesser values showed worse results.

The expression for calculating the gradient of this function is as follows:

$$\nabla R_{TV}(c(x)) = \nabla \cdot \frac{\nabla c}{\sqrt{\varepsilon^2 + ||\nabla c(x)||^2}} \quad (8)$$

As mentioned earlier, the calculation of the gradient by the conjugate state method is complicated by the need to read and process large amounts of data at each iteration, which with restrictions on the input-output system significantly increases the time for the so-called “overhead”. To accelerate data inversion process, the method of ordered subsets was used [Hud94]. This method consists in dividing the whole sensor data into several sets, each of which contains a certain number of frames. For each such set, an iteration step is performed that minimizes some local error function defined separately for each set. In order for the minimizing algorithm to work correctly for small subsets, the error function was normalized with respect to the real data of the frames of each subset. As a result, much smaller amount of data is required for each iteration, which significantly speeds up the process without noticable quality reduction of the final image.

4. EXPERIMENTAL RESULTS

Numerical modeling

The software application for the proposed method was implemented in MATLAB medium with the k-Wave Toolbox [Tre14], which allows to model a forward problem solution, including ultrasound frequency range.

Sequential algorithms for solving the forward and inverse problems are computationally expensive: the steps of finding the direction and optimization step requires an additional solution of 4-5 forward problems. At the stage of calculating the gradient by the conjugate state method, it is required to preserve the amplitudes not only at the locations of the transducers, but also in the rest of the region, at every moment in time. The volume of data to be processed at each iteration for the mentioned model is more than 100 GB. Data of such a size cannot be stored in RAM, therefore, when performing the necessary calculations, an intensive disk exchange operations occur.

The resource-intensive process of solving the forward problem was optimized by parallel computations on the GPU using CUDA technology, as well as the MATLAB Parallel Computing Toolbox. As a result of such optimization the computation speed of the program increased by approximately one order.

Due to the loaded computer resources, the program was tested on a small but acoustically representative model as a vertical cross-section of underwater pipeline filled with motor oil.

On Fig.1 the modeled pipeline is presented in the form of sound speed distribution in [m/s] and contains a foreign object of dense material. The size of the model in X and Y dimensions are 138x128 respectively in steps of computational grid for numerical solution of the equation (1). Linear size of each step is dx=dy=0.132 mm. The tube diameter is equal to 40 steps.

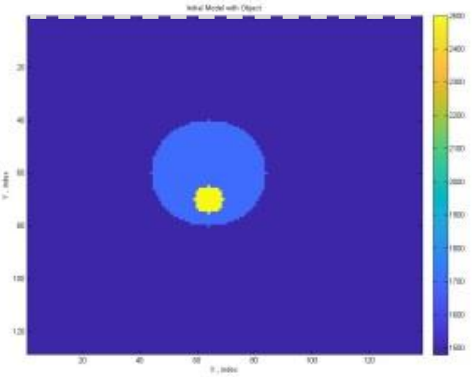


Fig.1. Initial model of underwater pipe cross-section, containing undesired object inside.

The model was probed with a single cycle pulse at a central frequency of 2 MHz. The array of 47 transducers is positioned along the X axis above the pipe at the upper margin of the model and is presented with a dash line. The distance between sensors is equal to 3 steps. The thickness of tube wall was considered to be much less than the wavelength and was not taken into account.

The decreasing dependency of the error function on the number of iterations in solving the inverse problem was experimentally established, which confirms the correctness of the optimizing algorithm. At 45 iterations the error function dropped down to the value of 2.3 × 10–7 (Fig.2).

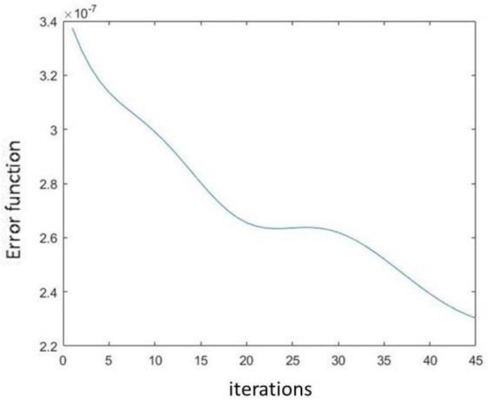


Fig.2. Error function dependency on the number of iterations.

During the inversion process the main object appears after the first iteration (Fig.3a), and the shape of small internal foreign object appears with an increase in the number of iterations (Fig.3b). Average number of iterations needed to achieve good image quality is equal to 4. Internal object round shape is easily identifiable, although sub-vertical boundaries of the pipe are weaker than sub-horizontal ones on both restored models. This is due to sub-vertical angles of incident waves that do not produce reflections.

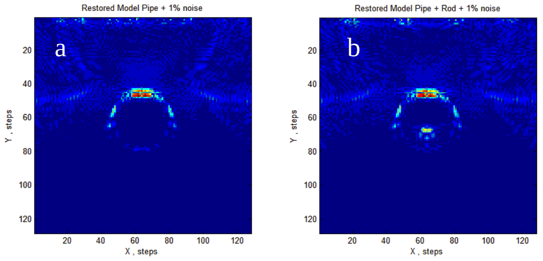


Fig.3. Reconstructed initial model without (a) and with internal object (b) with 1% of added noise.

The proposed method is quite resistant to the level of noise in the modeled wave field. With an increase in the level of additive noise from 1% (Fig. 3) to 10% (Fig. 4) the configuration of the pipe as well as the shape of internal object can be traced, although the lower wall of the tube is much less visible. With a further increase in the noise level, the quality of the reconstructed images degrades to unsatisfactory level, which imposes certain requirements to the development of acquisition equipment for non-destructive inspection based on tomographic images.

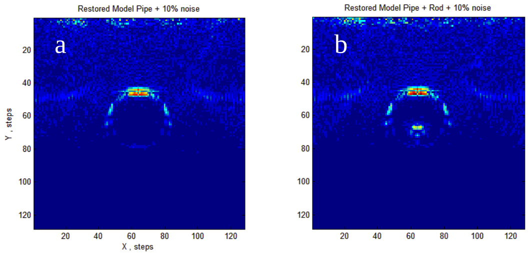


Fig.4. Reconstructed models with level of added random noise increased to 10%.

The shape of initial pulse complicates the reconstruction result and requires sophisticated deconvolution methods, which are not a subject of this paper. Fig.5 shows the result of model reconstruction by the proposed method at source frequency 2 MHz with sounding pulse having three and five cycles.

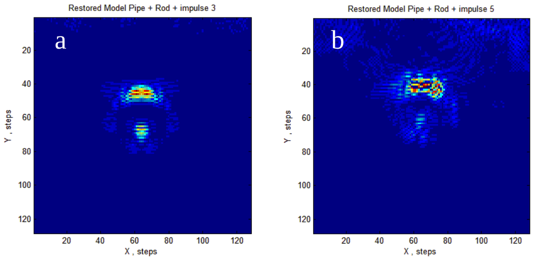


Fig.5. Restored model containing pipe and internal object with different shape of initial pulse: (a) – three cycles, (b) five cycles.

It is noticeable that more complex shape of the pulse leads to defocusing of the image and its content could be misinterpreted. Overall resolution of the implemented waveform inversion is about few grid steps, which is about half of wavelength benchmark. It is a little better in the tomographic approach compare to regular US techniques due to insonification of the medium by a vast number of waves at different angles.

For the selected model and acquisition geometry the solution of one forward problem on personal computer with 4 cores, 2.6 GHz, 8 GB RAM, 500 GB HDD and with graphics card NVIDIA GeForce-RTXsuper2060 takes less than one minute. The inverse problem is solved in about 6 minutes.

Physical modeling

Physical model was designed based on numerical model magnified by scale 10:1 since the sounding pulse of real transducers has center frequency 10 times lower, which is about 200KHz (Fig.6). The sizes of water tank were chosen equal to *L*x*W*x*D*=70x40x30 cm. In the first experiment, the thin plastic tube 40 cm long and 40 mm in diameter was filled with oil and was placed horizontally under the water at position 0.55*L* in *W* direction and at depth *D* equal to 40 mm. In the second experiment, a round ceramic rod 10 mm in diameter and 40 cm long was placed inside the tube at position ¾ of its diameter below the center, as shown in Fig.1.

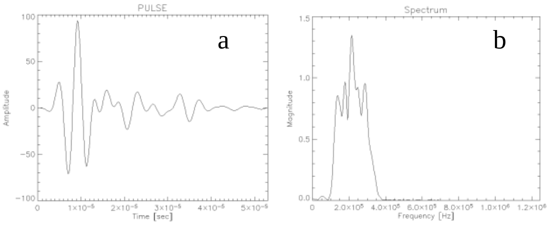


Fig.6. Source pulse (a) and its spectrum (b).

The 23 cm long array of 47 transducers was placed in the middle of water tank along the *L* direction with its center position [0.5*L*, 0.5*W*]. This acquisition geometry was implemented by means of precise automized positioning equipment. Due to very stable source pulse, only two transducers were used as a source-receiver pair and the tomographic sensor array was emulated the following way. One transducer served as emitter-receiver and moved consequently through all 47 positions with 5 mm distance between them. Second transducer served as receiver and for each position of the source it moved through the rest 46 positions. The used transducers have very broad directionality chart, which allowed to consider the registered reflection field to be generated by quasi-point source. The spatial sizes of tank provided enough space to avoid any side or bottom reflections within the time of recoding the primary wave signals, which was equal to *T*=270 μs.

Finally, a set of 47 frames with 47 records each was received. The set was preprocessed and used as an input for inverse problem solution. The results of experimental data inversion is presented at Fig.7.

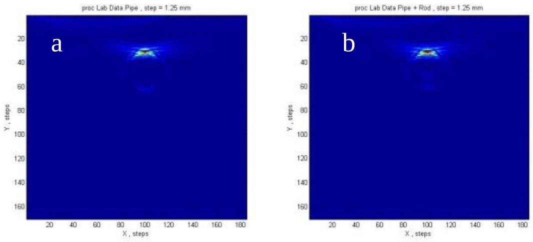


Fig.7. Reconstructed images of physical models of the underwater pipe (a), containing internal foreign object (b).

The quality of the restored images is not as good as on the numerical models described above due to the complexity of the sounding pulse, possible spatial inaccuracy in transducer positioning and level of real noise during the acquisition process. Also not all nonlinear effects are taking in account in the implemented software package. A simple thresholding procedures were applied to balance the contrast of the restored images. Some post processing efforts need be applied to suppress the noise in the final image and to make it more suitable for interpretation and decision making, but this is beyond the content of this paper. Nevertheless the shapes of the pipe and the inside object can be recognized and correct position and size of both physical objects can be retrieved.

5. CONCLUSION

The original algorithm was proposed for 2-D tomographic ultrasound images reconstruction based on waveform inversion of the reflected field in time domain. The proposed algorithm featured accuracy and computational efficiency and is more resistant to noise compare to finite-difference methods, however, it must be improved to meet the class of real noise in physical models and real structures. The advantage of the method is that there are no restrictions on the size of the inspected media because reflected waves can be recorded on its surface or at a distance without physical contact between sensor array and the targeted object. The resolution of tomographic images is higher than in traditional NDI approaches due to the multiple number of wave propagated through the medium at various angles. In order to obtain good image quality of the studied objects, it is necessary to select the optimal sounding frequency and shape of the emitted pulse. A volumetric image can be represented by a set of two-dimensional slices. These advantages will allow the proposed approach to be applied to a wider range of non-destructive testing and inspection tasks. The disadvantage of the approach on reflected waves may be the absence of reflections from the vertical boundaries of the studied objects, which may impose certain requirements on the size and geometry of the acquisition system with appropriate number of channels. The developed algorithm was implemented using GPU parallel computations, which significantly reduced the time of image reconstruction. The simplification of the spatial model of the acoustic medium consisting of objects of a relatively simple shape with nominal parameters, does not fundamentally affect the solution of the problem and is due only to computing power.

The results of numerical and physical modeling can be scaled by wavelength to achieve the required resolution in the infrastructural objects of the real size. For example, underwater oil pipeline of 1 m in diameter and with a wall thickness of 10 mm or less can be inspected without physical contact with transducers functioning at ultrasound frequency about 50 KHz with resolution inside the pipe about 13 mm.

The solution of the problem in 3-D is of further interest for obtaining volumetric high-resolution tomographic ultrasound images for non-destructive inspection of complex structures and equipment.

6. REFERENCES

[Bel17] Belykh I., Markov D. Volumetric ultrasound imaging: modeling of waveform inversion // Computer Science Research Notes 2703, 2017, P. 77–82.

[Bli96] Blitz, J. and Simpson, G. Ultrasonic Methods of Non-Destructive Testing, 1996.

[Gon17] Goncharsky A.V., Romanov S.Y., Seryozhnikov S.Y. Inverse Problems of Layer-by-Layer Ultrasonic Tomography with the Data Measured on a Cylindrical Surface. // Vychislitelnyye metody i programmirovaniye. MGU, 2017, T. 18, P. 267–276, (rus).

[Hua16] Huang L., Shin J., Chen T., Lin Y., Gao K., Intrator M., Hanson K. Breast ultrasound tomography with two parallel transducer arrays // Proc. SPIE 9783, Medical Imaging 2016: Physics of Medical Imaging, 97830C-1, 2016, 12 p.

[Hud94] Hudson H. M., Larkin R. S. Accelerated image reconstruction using ordered subsets of projection data // IEEE Trans. Med. Imaging, 1994, P.100–108.

[Li08] Li C., Duric N., Huang L. Clinical breast imaging using soundspeed reconstructions of ultrasound tomography data // Medical Imaging Int. Society for Optics and Photonics, 2008, P. 692009-1–692009-9.

[Lin12] Lin Y., Huang L., Zhang Z. Ultrasound waveform tomography with the total-variation regularization for detection of small breast tumors // SPIE Medical Imaging, 2012, P. 832002-1–832002-9.

[Liu17] Liu Y., Teng J., Xu T., Badal J., Liu Q., Zhou B. Effects of conjugate gradient methods and step-length formulas on the multiscale full waveform inversion in time domain: numerical experiments // Pure and Applied Geophysics, 2017, P. 1983–2006.

[San15] Sandhu G.Y., Li C., Roy O., Schmidt S., Duric N. Frequency domain ultrasound waveform tomography: breast imaging using a ring transducer // Physics in Medicine and Biology, 2015, Vol. 60, no. 14, P. 5381–5398.

[Sei17] Seidl R. Full Waveform Inversion for Ultrasonic Nondestructive Testing. PhD thesis, TECHNISCHE UNIVERSITAET MUNCHEN, 2017.

[Tre14] Treeby B. E., Jaros J., Rohrbach D., Cox B. T. \Modelling Elastic Wave Propagation Using the k-Wave MATLAB Toolbox, in IEEE Int. Ultrasonics Symposium, pp. 146-149, 2014.

[Ver18] Vereshchagin K.A., Belykh I.N. Ultrasound image reconstruction based on waveform inversion method using parallel calculation algorithms. // St. Petersburg State Polytechnic University Journal: Computer Science. Tele-communications and Control Systems. Vol. 11, No. 4, 2018, P. 49-62, (rus).

[Vir09] Virieux J., Operto S. An overview of full waveform inversion in exploration geophysics // Geophysics, 2009, P. WCC1–WCC26.

[Zie18] Zielińska M., Rucka M. Non-Destructive Assessment of Masonry Pillars using Ultrasonic Tomography. Materials 2018, 11, 2543; doi: 10.3390/ma11122543.

A Flexible Pipeline for the Optimization of CSG Trees

Markus Friedrich, Christoph Roch, Sebastian Feld,
Carsten Hahn
Institute for Computer Science
LMU Munich
Oettingenstr. 67
80538 Munich, Germany
{first name.last name}@ifi.lmu.de

Pierre-Alain Fayolle
Division of Information and Systems
The University of Aizu
Aizu-Wakamatsu City
965-8580 Fukushima, Japan
fayolle@u-aizu.ac.jp

ABSTRACT

CSG trees are an intuitive, yet powerful technique for the representation of geometry using a combination of Boolean set-operations and geometric primitives. In general, there exists an infinite number of trees all describing the same 3D solid. However, some trees are optimal regarding the number of used operations, their shape or other attributes, like their suitability for intuitive, human-controlled editing. In this paper, we present a systematic comparison of newly developed and existing tree optimization methods and propose a flexible processing pipeline with a focus on tree editability. The pipeline uses a redundancy removal and decomposition stage for complexity reduction and different (meta-)heuristics for remaining tree optimization. We also introduce a new quantitative measure for CSG tree editability and show how it can be used as a constraint in the optimization process.

Keywords
Evolutionary Algorithms, Quantum Annealing, Geometry Processing, CAD, CSG, Combinatorial Optimization

1 INTRODUCTION

Constructive Solid Geometry (CSG) trees are a powerful representation scheme for 3D geometry and an important building-block of 3D modelling software [Req80]. While the creation, reconstruction and conversion from other representations of CSG trees has been covered in the literature, few works deal with the optimization of a given tree.

The nature of CSG tree expressions as a combination of geometric primitives (so-called halfspaces) with Boolean set-operations (union, intersection, complement) suggests a deeper investigation of related methods from *switching function minimization*. This paper investigates an adaption of already proposed, as well, as newly developed methods for the CSG tree optimization problem and extends them to a robust and flexible pipeline. Furthermore, it introduces the idea that CSG tree optimization does not have to be restricted to the reduction of the tree size but should include improvements to the tree's *editability*.

We consider our CSG tree optimization pipeline as a solution to the following problems: Given a hand-modeled CSG tree with sufficient complexity, users have difficulty keeping track of potentially redundant parts. An automatic, but manually triggered, optimization procedure comes in handy. Furthermore, automatic CSG tree reconstruction methods [FP16, WXW18] might result in trees that are not optimal in size and hard to edit manually. Thus, our method can be beneficial in this scenario as well.

This paper makes the following contributions:

- The description of a comprehensive pipeline for optimizing the *editability* of a CSG expression,
- A novel, sampling-based tree size optimization procedure suitable for Quantum Annealing hardware,
- A recursively defined measure of spatial subtree proximity as an indicator for CSG tree *editability*,
- A multi-objective optimization using a Genetic Algorithm that aims to minimize the CSG expression size and maximize its proximity value in order to improve tree *editability*.

The paper is organized as follows: Section 3 gives basic definitions and concepts. Section 2 provides references to related works, whereas Section 4 defines the problem to solve. Our approach is described in Section 5 and evaluated in Section 6. Section 7 concludes the paper.

2 RELATED WORK

Construction of CSG trees from the Boundary Representation (B-Rep) of a solid was considered by Shapiro and Vossler in [SV91a, SV91b, SV93]. The approach is

https://doi.org/10.24132/CSRN.2020.3001.10

based on: identifying a set of halfspaces sufficient for representing the input solid, building a CSG expression by considering all products of halfspaces, or their complement, that are inside the input solid (so-called fundamental products), and minimizing this expression. In particular, [SV91a] considers different approaches for the minimization of two-level expressions, either coming from switching theory [Qui52, MJ56, O'R82], or based on geometric considerations.

Related to these works, Buchele and Crawford [BC04] propose an algorithm for producing a CSG expression from the Boundary Representation of a solid by considering early factoring of dominant halfspaces. Such a factoring should help in limiting the size of the produced CSG expression. Andrews proposes to simplify a CSG expression (obtained from a B-Rep) by removing from each fundamental product cell, (spatially) distant primitives [And13]. The reason is that distant primitives may be viewed by the user as unrelated to a given fundamental product cell, and thus could lead to unintuitive results.

Recently, works on reconstructing a CSG expression from a point-cloud have become popular, see for example [FP16, WXW18] , among others. The approach described in [FP16] tries to minimize the size of the CSG expressions produced by a GA by penalizing large expressions in the objective function. The method proposed in [FFGLP19] is using multi-objective optimization to prevent the growth of the generated CSG expressions. It also uses a decomposition scheme that prevents spatially distant primitives to be used in unrelated CSG sub-expressions.

Other works related to the optimization, or manipulation, of CSG expressions, such as [Ros11], try to improve the rendering time of the model, and do not necessarily help in minimizing the size of the expression or improving its editability.

3 BACKGROUND

3.1 CSG Tree Representation

3.1.1 Formal Representation

We follow the formal definition of CSG trees from Shapiro et al. [SV91a]: Given a solid's point-set S, its boundary ∂S consists of patches of halfspaces H_S. Halfspaces are regular sub-sets of the universal point-set W usually described by signed distance functions (SDF) F_H: $\{\mathrm{x} \in \mathbb{R}^3 : F_H(\mathrm{x}) = 0\}$. A CSG tree expression Φ (in the following, upper-case Greek letters are used for CSG tree expressions) consists of halfspace literals $\{h_0, h_1, ...\}$ and symbols for regularized set-operations $\{\cup^*, \cap^*, \backslash^*, -^*\}$. Applying Φ to the set of halfspaces H_S results in a CSG representation $\Phi(H_S)$ of S iff $|\Phi(H_s)| = S$, where $|\cdot|$ denotes the point-set induced by a CSG representation.

A CSG representation of S is in disjunctive normal form (DNF) if it contains a sum ($\cup^*$) of halfspace products ($\cap^*$-combined halfspaces or negated halfspaces, so-called implicants). If each implicant of a DNF expression contains all halfspaces (or their negations), it is in a so-called disjunctive canonical form (DCF). In that case, an implicant is called canonical intersection term (CIT) or fundamental product (FP). For example, the universal set W can be decomposed in 2^n CITs in case of n halfspaces being used.

An implicant Ψ is a so-called prime implicant of S if $|\Psi| \in S$ and the removal of a single halfspace from Ψ results in $|\Psi| \notin S$. A dominant halfspace (DH) $g \in H_S$ is a halfspace for which $S = g \cup^* S$ is always true (g is then also a prime implicant of S).

Where necessary, we use the following abbreviations for CSG expressions (halfspaces and set-operations): $|h_0| \cup^* |h_1| := h_0 + h_1$, $|h_0| \cap^* |h_1| := h_0 \cdot h_1$, $\backslash^* |h_0| := \overline{h_0}$ and $|h_0| -^* |h_1| := h_0 - h_1$.

3.1.2 SDF-based implementation

In order to compute the point-set of a CSG representation (e.g., $|\Phi(H_s)|$) we use an SDF-based approximation with min- and max-functions [Ric73, PASS95, Sha07]:

- Intersection: $|\Phi| \cap^* |\Psi| := \max(F_\Phi, F_\Psi)$
- Union: $|\Phi| \cup^* |\Psi| := \min(F_\Phi, F_\Psi)$
- Complement: $\backslash^* |\Phi| := -F_\Phi$
- Difference: $|\Phi| -^* |\Psi| := \max(F_\Phi, -F_\Psi)$

Here, F_Φ and F_Ψ are the SDFs corresponding to expression Φ and Ψ. We assume, as a convention, that $F < 0$ in the interior of the corresponding solid S.

Note that min- and max-functions are not regularized set-operations in the strict sense [Sha99] but a sufficient approximation for our purposes.

3.2 Metaheuristics for Combinatorial Problems

3.2.1 Genetic Algorithms

Genetic Algorithms (GA) are metaheuristics for solving discrete or continuous optimization problems. The process is inspired by biology and consists of evolving a population of creatures. Each creature represents a candidate solution to the problem. Starting from a randomly initialized population, a GA produces an updated population at each iteration by: a) ranking creatures according to a fitness function, b) generating new creatures by mutating a selected creature from the previous population, or by combining a selected pair of creatures from the previous population, c) selecting a few creatures to be preserved into the next population (elitism). Selection is performed based on the rank of each creature. The process is iteratively repeated until a termination criterion is met.

3.2.2 *Quantum Annealing*

Quantum Annealing (QA) is another metaheuristic for solving (in general, discrete) optimization problems. It is based on quantum physics to find low energy states of a system corresponding to the optimal solution of a problem. The QA algorithm is described by a time-dependent Hamiltonian $\mathscr{H}(t)$:

$$\mathscr{H}(t) = s(t)\mathscr{H}_I + (1-s(t))\mathscr{H}_P$$

The QA process starts in the lowest-energy state of a so-called initial Hamiltonian $\mathscr{H}_I$. During the annealing process, the problem Hamiltonian $\mathscr{H}_P$ is introduced and the influence of the initial Hamiltonian is reduced (described by $s(t)$, which decreases from 1 to 0). At the end of the annealing process, one ends up in an eigenstate of the problem Hamiltonian, which actually encodes the objective function of the problem. If this transition is executed sufficiently slowly, the probability to find the lowest energy state of the problem Hamiltonian is close to 1, w.r.t the adiabatic theorem [AL18].
To perform QA on D-Wave Systems Quantum Annealing hardware, one needs to encode the problem ($\mathscr{H}_P$) in a so-called Quadratic Unconstrained Binary Optimization (QUBO) problem, which is a unifying model for representing a wide range of combinatorial optimization problems. The functional form of the QUBO the quantum annealer is designed to minimize is:

$$\min \mathrm{x}^T Q \mathrm{x} \quad \text{with } \mathrm{x} \in \{0,1\}^{n_Q}, \tag{1}$$

where x is a vector of binary variables of size n_Q, and Q is an $n_Q \times n_Q$ real-valued matrix describing the relationship between the variables. Given the matrix Q, the annealing process tries to find binary variable assignments to minimize the objective function (Eq. 1).

4 PROBLEM STATEMENT

We focus on the optimization of a CSG tree's *editability*: Given a solid's point-set S, a halfspace set H_S and a CSG tree expression Φ with $|\Phi(H_S)| = S$, find the CSG tree expression Φ_{opt} with the best *editability* which is assumed to be determined by two quantitative metrics:

- **Size:** The amount of literals and operations in Φ_{opt}.
- **Proximity:** The ratio between the number of operations of Φ_{opt} whose operands imply point-sets that overlap to the number of operations with operands that imply disjoint point-sets. This property is defined recursively as follows: Given a node Ψ of Φ_{opt}, either Ψ is a leaf (an halfspace) or it has two children (operands) Ψ_1 and Ψ_2 such that the implied solids intersect (that is: $|\Psi_1| \cap^* |\Psi_2| \neq \varnothing$).

A size-optimal tree has no redundant operands which makes tree modification easier and a tree with a high degree of proximity leads to more predictable behavior when sub-trees are transformed spatially.

5 CONCEPT

The CSG tree optimization process is depicted in Fig. 1: First, redundant sub-expressions are removed (orange, Section 5.1). Then, a recursive decomposition scheme is applied that further shrinks the expression size (grey, Section 5.2). If an unoptimized expression (solid) remains, it is optimized with a separate optimization method (blue or purple, Section 5.3) which results – after another run of the redundancy removal method – in the final optimized CSG tree.

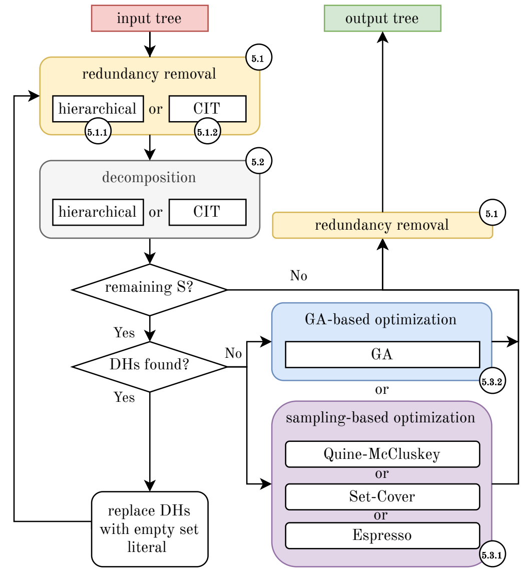


Figure 1: The proposed CSG tree optimization pipeline.

5.1 Redundancy Removal

Our redundancy removal approach is inspired by a method described in [Til84]. It uses the spatial information additionally given by the halfspaces used in the CSG tree (e.g., a sphere's location and radius):

- If the sets described by the operands (halfspaces or subtrees) of an intersection operation do not have elements in common, i.e., the operand sets do not spatially overlap, the expression is replaced with the empty set expression $\varnothing$.
- If the sets described by the operands of a union operation are identical, the expression is replaced with one of the operand expressions.

Empty set $\varnothing$ as well as universal set W expressions are then replaced based on the following rules:

- If one operand of an intersection expression is the empty set $\varnothing$, the expression is replaced with the empty set.
- If one operand of an intersection expression is the universal set W, the expression is replaced with the other operand.
- If one operand of a union expression is the empty set $\varnothing$, the expression is replaced with the other operand.

- If one operand of a union expression is the universal set W, the expression is replaced with W.

In addition, the complement of a complement operation is replaced with the operand of the inner complement operation. The redundancy removal algorithm continuously iterates over the whole CSG tree until no rule applies anymore to the current result.

Especially relevant for the algorithm is the fast and robust evaluation of the empty set and identical set decision algorithms. Since the identical set decision can be expressed as an empty set decision ($|\Phi| = |\Psi| \iff |\Phi| \cap^* (\backslash^* |\Psi|) = \varnothing \wedge |\Psi| \cap^* (\backslash^* |\Phi|) = \varnothing$), an empty set decision algorithm is sufficient.

We use a sampling-based approach: If the tree represents an empty set, its SDF value is positive (*outside*) in the complete sampling domain. Two different sampling strategies are proposed: hierarchical sampling and CIT-based sampling.

5.1.1 Hierarchical Sampling

This sampling method is an Octree-based, hierarchical sampling of the CSG tree's SDF. The sampling point-set is defined by the width, height and depth of the axis-aligned bounding box (AABB) dimensions of the used halfspaces (w_0, h_0, r_0) and the user-defined minimum sampling cell size $(w_{min}, h_{min}, r_{min})$. The coarse-to-fine hierarchical sampling methodology, as depicted in Fig. 2 allows for early stopping in case of a non-positive SDF value. To further speed-up the process, a lookup-table for already proven empty set expressions is used.

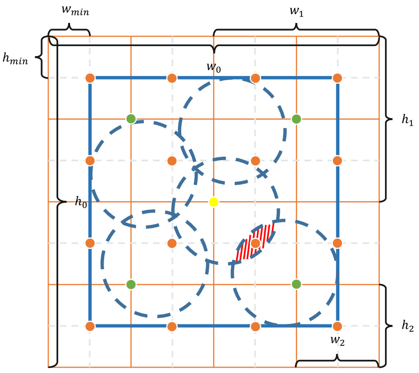


Figure 2: Hierarchical sampling strategy. The space defined by (w_0, h_0, r_0) is consecutively subdivided in subquadrants of sizes $s_i = (\frac{w_{i-1}}{2}, \frac{h_{i-1}}{2}, \frac{r_{i-1}}{2}), i = 1, 2, \ldots$ until $w_i \leq w_{min} \vee h_i \leq h_{min} \vee r_i \leq r_{min}$. The center of each quadrant is the sampling position (yellow, green, orange). In the example, early stopping is possible since the red-hatched area, which marks the non-empty volume, is sampled at quadrant size (w_2, h_2, r_2) which is one level above the smallest possible quadrant size.

5.1.2 CIT-Based Sampling

This sampling method uses a sampling point-set consisting of a point for each CIT located inside the tree. CITs are retrieved with the method explained in Section 5.3.1. For each point we check if the tree's SDF is positive. The approach can potentially be faster than hierarchical sampling since the used sampling point-set is usually smaller.

5.2 Decomposition

The decomposition of a solid S (that is described by a CSG tree Φ and halfspaces H_S) as proposed in [SV91a] is a tree expression that consists of a chain of halfspaces from H_S that either dominate S or $\overline{S}$ and a (potentially empty) remaining solid S_{rem}:

$$S = |((...(S_{rem} \oplus d_1) \oplus ...) \oplus d_2) \oplus d_n|, \qquad (2)$$

where $\{d_1, ..., d_n\}$ is the set of dominating halfspaces and $\oplus$ is either $+$ if the following halfspace dominates S or $-$ if it dominates $\overline{S}$. This decomposition is a size-optimal tree expression for S since each halfspace appears only once [SV91a]. For example, in Fig. 3, h_0, h_2 dominate $\overline{S}$ and h_3 dominates S, resulting in the decomposition $S = |((h_4 - h_0) - h_2) + h_3|$. Here, S_{rem} contains h_4 and h_1 but since h_1 has no impact on the result, S_{rem} equals h_4 and thus is replaced in the result expression. Please note that we use the example in Fig. 3 differently than in Section 5.3.1: The solid shown is considered S, which is decomposed to get S_{rem}, while in Section 5.3.1 the solid shown is considered S_{rem} which is then further optimized.

Recursive Decomposition. If S_{rem} is not empty, a size-optimal expression for it has to be found. Therefore, a (potentially not size-optimal) expression Φ_{rem} is computed by replacing all appearances of previously found dominating halfspaces in Φ with the empty set literal and applying the redundancy removal method (see Section 5.1). This is followed by another run of the decomposition technique. This recursive process is continued until either the current S_{rem} is empty, or no more dominating halfspaces can be found during decomposition. In the latter case, an optimal expression for S_{rem} has to be found using different methods, see Section 5.3.

Sampling-Based Search for Dominant Halfspaces. For the identification of the dominant halfspaces of S, we propose two different sampling strategies as already described in Section 5.1: Firstly, a hierarchical sampling strategy similar to that proposed in Section 5.1.1 can be used. But instead of taking the whole AABB of S as sampling volume, each halfspace h in H_S is separately tested using its corresponding AABB. If all sampling points inside $|h|$ are inside S as well, h dominates S. If all sampling points inside $|h|$ are not elements of S, h dominates $\overline{S}$. The early-stopping criteria is met if a sampling point is in $|h|$ but not in S. Secondly, a CIT-based sampling strategy as discussed in Section 5.1.2 can be used.

Improving Proximity. Although decomposition can result in size-optimal expressions, the *editability* criterion is also influenced by the proximity metric which is not considered by the proposed decomposition method. To overcome this deficiency, we propose a simple spatial sorting scheme: The halfspaces in the chain of halfspaces (Eq. 2) are arranged such that operands of $\oplus$ do always spatially overlap (if possible).

5.3 Remaining Solid Optimization

After decomposition, there might be a remaining solid $S_{rem} = |\Phi_{rem}(H_{rem})|$ left, for which an optimal expression has to be found. We investigate two optimization methods: Firstly, a sampling-based technique that generates tree expressions in DCF form that are then optimized using well-known two-level logic minimization techniques (Quine-McCluskey [MJ56], Espresso logic minimizer [BHMSV84]) and a new approach based on a QUBO formulation of the Set Cover problem (Section 5.3.1). Secondly, a GA-based method is proposed that uses tree size and the proximity metric as part of its objective function (Section 5.3.2).

5.3.1 Sampling-Based Optimization

Sampling. The AABB of S_{rem} is sampled. For each sample, the tree's SDF is evaluated to decide if the point is *inside*. If it is *inside*, each halfspace SDF in H_{rem} is evaluated as well. If the sampling point is located *inside* the halfspace, the halfspace is part of a CIT. If not, then its complement is. Finally, the CIT is added with a $\cup^*$ operation to the resulting DCF expression Φ_{DCF}. See Fig. 3 for an example.

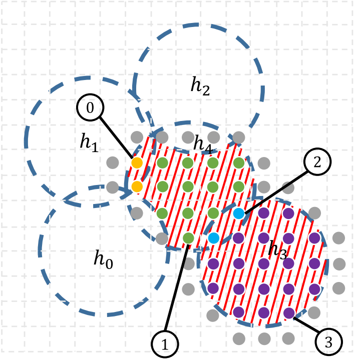


Figure 3: Example of the sampling step. The red-hatched area is the point-set to represent, $\{h_0, ..., h_4\}$ is the halfspace set H_{rem}. The grey dots indicate sampling points outside the solid. The orange dots result in the implicant $\overline{h_0} \cdot h_1 \cdot \overline{h_2 \cdot h_3} \cdot h_4$ (0), the green dots in $\overline{h_0} \cdot \overline{h_1 \cdot h_2 \cdot h_3} \cdot h_4$ (1), the light blue dots in $\overline{h_0} \cdot \overline{h_1 \cdot h_2} \cdot h_3 \cdot h_4$ (2) and the purple dots in $\overline{h_0} \cdot \overline{h_1 \cdot h_2} \cdot h_3 \cdot \overline{h_4}$ (3).

DCF Minimization. Besides the already mentioned classic methods for DNF minimization (Quine-McCluskey, Espresso logic minimizer), we propose a third option: First, the prime implicants of Φ_{DCF}, P_{DCF}, are computed by directly applying the definition of a prime implicant (see Section 3) to each CIT in Φ_{DCF}. Using the example of Fig. 3, the prime implicants would be h_3 and $\overline{h_0 \cdot h_2} \cdot h_4$ (Note that we do not consider decomposition for this example).

The problem of finding all relevant prime implicants is then formulated as a Set Cover problem: The set to cover, U, is the set of indices of all CITs located inside $|\Phi_{DCF}|$ (in Fig. 3: $U = \{0,1,2,3\}$). Each prime implicant in P_{DCF} covers a subset of U resulting in a collection of subsets V with elements $V_k \subseteq U, 1 \leq k \leq \text{card}(P_{DCF})$ (in Fig. 3: $V = \{h_3 : \{2,3\}, \overline{h_0 \cdot h_2} \cdot h_4 : \{0,1,2\}\}$). Within the Set Cover problem, one has to find the smallest possible number of subsets from V, such that their union is equal to U. This problem was proven to be NP-hard [Kar72]. In [Luc14] the QUBO formulation for the Set Cover problem is given by:

$$\mathscr{H}_A = A \sum_{\alpha=1}^{\text{card}(U)} \left(1 - \sum_{m=1}^{\text{card}(P_{DCF})} x_{\alpha,m}\right)^2 + A \sum_{\alpha=1}^{\text{card}(U)} \left(\sum_{m=1}^{\text{card}(P_{DCF})} m x_{\alpha,m} - \sum_{k:\alpha \in V_k} x_k\right)^2,$$

and

$$\mathscr{H}_B = B \sum_{k=1}^{\text{card}(P_{DCF})} x_k, \tag{3}$$

with x_k being a binary variable which is 1, if set V_k is included within the selected sets, and 0 otherwise. $x_{\alpha,m}$ denotes a binary variable which is 1 if the number of selected subsets V_k which include element α is $m \geq 1$, and 0 elseways. The first energy term in $\mathscr{H}_A$ imposes the constraints that for any given α exactly one $x_{\alpha,m}$ must be 1, since each element of U must be included a fixed number of times. The second term in $\mathscr{H}_A$ states that the number of times that we declared α was included is in fact equal to the number of subsets V_k we have included with α as an element. A is a penalty value, which is added on top of the solution energy, described by $\mathscr{H} = \mathscr{H}_A + \mathscr{H}_B$, if a constraint was not fulfilled, i.e., one of the two terms are unequal to 0. Therefore adding a penalty value states a solution as invalid. Additionally, the Set Cover problem minimizes over the number of chosen subsets V_k, as stated in (Eq. 3). For a given problem instance, $\mathscr{H}$ is transformed into the QUBO formulation required by QA hardware (Eq. 1) and minimized.

Note that the prime implicant selection via Set Cover is not needed if H_{rem} does not contain halfspaces that fully contain other halfspaces and no separating halfspaces [SV91a] are used. Furthermore, the sampling step can be omitted if the input expression is already in DNF form.

5.3.2 GA-Based Optimization

The methods described in Section 5.3.1 are two-level minimization techniques, which in general do not result in size-optimal trees [SV91a]. Moreover, other optimization goals like, for example, the proximity metric are not considered. Their main advantages are a possible short execution time and the fact that non-optimality of sufficiently small trees (e.g., after decomposition as explained in Section 5.2) has usually smaller and thus negligible negative effects on tree size and proximity.

In order to find possibly better trees (w.r.t. size and proximity), the corresponding problem is formulated as a combinatorial optimization problem over all possible trees given a set of halfspaces H_{rem} and set-operations. Let Φ_{rem} be the input CSG tree expression that needs to be optimized. A GA is used to solve this optimization problem. Each creature Φ_c in the population of the GA represents a potential CSG tree expression. The same mutation, crossover and selection as in [FP16] are used. Additional details about the GA are provided below.

Initialization. The population is initialized with a mixture of randomly generated trees with a maximum size equal to the size of the input tree Φ_{rem} and copies of the input tree.

Pre-processing. To reduce the computational effort of evaluating the fitness function in the GA, we compute a limited number of sample points. This is done by computing the CITs of Φ_{rem} via sampling as described in Section 5.3.1 with the difference that CITs that are located outside of Φ_{rem} are considered as well. During the sampling process, a point in each CIT is added to the sampling point-set S_{in} if the corresponding CIT is inside of Φ_{rem}. Otherwise, it is added to S_{out}.

Ranking. The fitness of a creature Φ_c corresponding to a given CSG tree is given by

$$f(\Phi_c) = \alpha \cdot f_{geo}(\Phi_c) + \beta \cdot f_{prox}(\Phi_c) + \gamma \cdot f_{size}(\Phi_c),$$

where $f_{geo}(), f_{prox}()$ and $f_{size}()$ are the geometric score, the proximity score and the size score, respectively, and α, β and γ are user-defined parameters. The geometric score counts how many points from S_{in} are elements of $|\Phi_c|$ and how many points from S_{out} are not in $|\Phi_c|$:

$$f_{geo}(\Phi_c) = f_{in}(\Phi_c) + f_{out}(\Phi_c),$$

with

$$f_{in}(\Phi_c) = \frac{1}{\text{card}(S_{in})} \sum_{s \in S_{in}} \begin{cases} 1, & \text{if } |F_{\Phi_c}(s)| \leq \varepsilon_p \\ 0, & \text{otherwise} \end{cases}$$

and

$$f_{out}(\Phi_c) = \frac{1}{\text{card}(S_{out})} \sum_{s \in S_{out}} \begin{cases} 1, & \text{if } |F_{\Phi_c}(s)| > \varepsilon_p \\ 0, & \text{otherwise} \end{cases},$$

where $\text{card}(S)$ is the cardinality of the point-set S.

The proximity score tries to enforce that two operands of a Boolean operation are spatially connected. It is an implementation of the proximity metric (Section 4):

$$f_{prox}(\Phi) = \frac{P_{rec}(\Phi)}{\#|\Phi|},$$

where $\#|\Phi|$ is the number of nodes in the tree corresponding to the creature Φ and the function

$$P_{rec}(\Phi) = \begin{cases} 1, \text{ if } \Phi \text{ is a leaf node} \\ P_{rec}(\Phi_1) + P_{rec}(\Phi_2) + \Delta(\Phi_1, \Phi_2) \text{ otherw.} \end{cases},$$

where Φ_1 and Φ_2 are the two children of Φ (if it is not a leaf node), and

$$\Delta(\Phi_1, \Phi_2) = \begin{cases} 1 \text{ if } |\Phi_1| \cap^* |\Phi_2| \neq \varnothing \\ 0 \text{ otherwise} \end{cases}.$$

Finally, the size score tries to favor the simplest (shortest) possible CSG tree and corresponds to

$$f_{size}(\Phi_c) = \frac{\#|\Phi_{rem}| - \#|\Phi_c| - f_{size}^{min}}{f_{size}^{max} - f_{size}^{min}},$$

where $\#|\Phi_{rem}|$ and $\#|\Phi_c|$ are the number of nodes in the CSG trees Φ_{rem} and Φ_c, f_{size}^{min} is the minimum and f_{size}^{max} the maximum tree size within the current iteration's population.

Termination. The GA terminates if either a user-defined maximum number of iterations is reached or the score does not improve over a number of iterations.

6 EVALUATION

6.1 Data Acquisition

We prepared 11 different, hand-crafted CAD models for our experiments (see Fig. 4) with a complexity comparable to models commonly used in CSG tree reconstruction tasks [FP16, DIP+18]. To simulate particular levels of *sub-optimality*, we have implemented a generator, which takes a CSG tree as input and iteratively adds redundant parts at random positions in the tree, based on the following strategies:

Copied Subtree Insertion (CSI): Insert a union or intersection at a random position with both operands being copies of the subtree at that position.

Double Negation Insertion (DNI): Insert two chained negations at a random position.

Distributive Law Insertion (DLI): Apply the distributive laws $A(B+C) = (A \cdot B) + (A \cdot C)$ or $A + (B \cdot C) = (A+B) \cdot (A+C)$ to a random subtree.

Absorption Law Insertion (ALI): Apply the absorption laws $A = A + (A \cdot B)$ or $A = A \cdot (A+B))$ to a random subtree.

GA-based Redundancy Insertion (GRI): Use the GA (Section 5.3.2) with negative size and proximity weight to produce trees with redundant parts.

A particular run of the generator can be parameterized by the tuple $(N_{\text{iter}}, P_{CSI}, P_{DNI}, P_{DLI}, P_{ALI}, P_{GRI})$, where N_{iter} is the number of iterations of the generator and $P_{...}$ are the probabilities of the insertion strategies. These artificially introduced redundancies cover all sorts of redundancies potentially appearing in CSG trees. Table

6.1 lists properties of the input models and their generated inflated versions. Data set 1 uses all possible inflation mechanisms with the same probability for 10 iterations $(10,1.0,1.0,1.0,1.0,1.0)$, data set 2 uses GA inflation only for 20 iterations $(20,0.0,0.0,0.0,0.0,1.0)$. All sampling techniques use a step size of 0.1.

6.2 Execution Times

For evaluating the execution time, we used a quad core CPU @2.80 GHz and 16GB of RAM. Set covers are computed without QA hardware acceleration.

6.2.1 Pipeline

For the pipeline steps *Redundancy Removal* and *Decomposition*, different options are possible: a) decomposition or b) the redundancy removal method can use the hierarchical sampling strategy (0) or CIT-based sampling (1) and c) redundancy removal can be used (1) or not (0). Options a) - c) can be combined resulting in 6 possible configurations. A particular configuration is identified by a binary 3-tuple, e.g., $(1,0,1)$ for decomposition with CIT-based sampling and redundancy removal with hierarchical sampling. All cases use Espresso for the remaining solid optimizer. Results are given in Fig. 5 and 6. For both data sets, CIT-based sampling is inferior in any configuration due to its processing time. The initial redundancy removal does not have any positive impact on execution times as well (exception: model 3 and model 10 in both data sets).

6.2.2 Remaining Solid Optimization

Results for data set 1 are given in Table 6.2.2. For data set 2 results are similar and thus are not shown due to space constraints. Overall the fastest method is Espresso, followed by the Quine-McCluskey method (exception: model 10 where Set Cover is faster). Within the Set Cover method, the prime implicant generation part is the most time expensive whereas solving the Set Cover problem is neglectable ($\sim$ 1-3ms). The slowest method is the GA-based optimization with the exception of model 8, where Set Cover is the slowest.

6.3 Optimization Characteristics

All experiments use the pipeline configuration $(0,0,1)$.

6.3.1 Pipeline

For both data sets, resulting size (Fig. 7) and proximity (Fig. 8) for models 1, 3, 4, 5, 6, 7 and 9 are the same for all methods since for these models, the remaining solids are empty after decomposition and thus no remaining solid optimization is necessary.

For both data sets, the GA produces the best size results (exception: model 11) but not always the best proximity results (e.g., data set 2, model 2). This can be explained by two factors: The GA's objective function's size weight used in our experiments is greater than its proximity weight (10 vs. 1) and larger trees have more redundancies and thus tend to have higher subtree overlap resulting in higher proximity scores. This explains as well the results of the Set Cover method which are the worst in terms of tree size but among the best regarding proximity. Compared to the original, handcrafted input trees, resulting trees have equally good or better size and proximity values for both data sets (exception: size for model 8, data set 2). Also worth mentioning is that the Quine-McCluskey method cannot handle the size of model 8.

6.3.2 GA Specifics

For the GA, we used a population size of 150, a mutation rate of 0.3, a crossover rate of 0.4 and a tournament selection $k = 2$. Objective function weights are $\alpha = 50, \beta = 1, \gamma = 10$. The maximum iteration count is 1000, and after 500 iteration without score change, the GA is terminated as well. Fig. 9 shows one of the main advantages of the GA approach for remaining solid optimization: It is possible to manually select the best size/proximity trade-off solution for a particular use-case after the optimization process has finished.

7 CONCLUSION

In this paper, we proposed a flexible new pipeline for the efficient optimization of a CSG tree's editability. We have evaluated different pipeline combinations with a representative set of models.

It is possible to use the sampling-based optimization method (Section 5.3.1) in situations where the input expression is not known in advance (only the solid point- and halfspace-set). In that case, another strategy for *inside-outside* decisions would be needed.

Furthermore, the decomposition technique (Section 5.2) can be also used for the task of CSG tree reconstruction from other geometry representations (e.g., point clouds or B-Rep representations). Therefore, the dominant halfspace detection must be adapted according to the geometry representation of the input and the discussed recursion would end after the first iteration without applied redundancy removal (which is not possible without a given input tree). A GA-based approach [FP16] (or the aforementioned variant of the sampling-based optimization method) could be used to find a tree expression for the remaining solid.

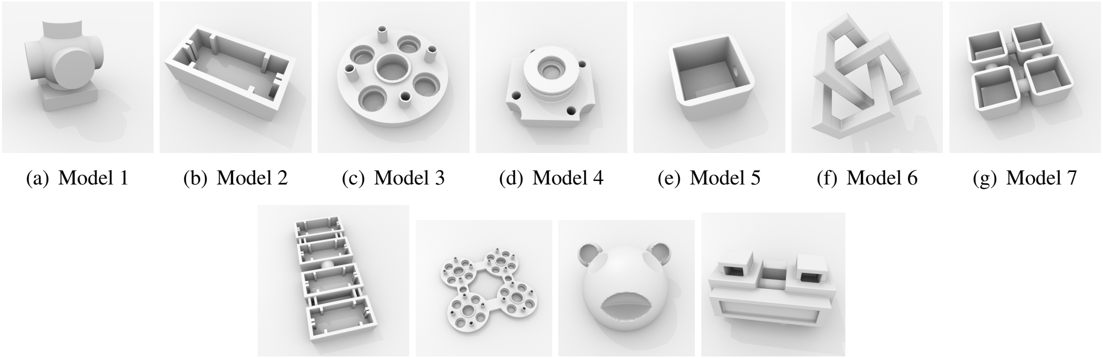
8 REFERENCES

[AL18] Tameem Albash and Daniel A Lidar. Adiabatic quantum computation. *Reviews of Modern Physics*, 90(1):015002, 2018.

[And13] James L Andrews. *User-Guided Inverse 3D Modeling*. PhD thesis, University of California, Berkeley, 2013.

Models (number of nodes, proximity score, (dimensions))	Data set 1	Data set 2
model1 (9, 0.75, (20.0×22.0×20.0))	(97, 0.442)	(78, 0.35)
model2 (13, 0.833, (15.8×31.7×11.3))	(41, 0.952)	(23, 0.727)
model3 (39, 0.474, (21.0×6.0×21.0))	(79, 0.675)	(57, 0.536)
model4 (27, 1, (13.4×13.4×12.0))	(65, 0.97)	(97, 0.708)
model5 (19, 0.667, (24.0×18.0×27.0))	(59, 0.645)	(48, 0.76)
model6 (19, 0.556, (23.1×10.0×10.0))	(99, 0.667)	(43, 0.857)
model7 (91, 0.706, (21.6×7.4×21.8))	(144, 0.725)	(151, 0.728)
model8 (73, 0.722, (31.5×12.7×4.5))	(145, 0.74)	(97, 0.813)
model9 (171, 0.471, (29.7×3.84×30.1))	(191, 0.51)	(216, 0.519)
model10 (17, 0.875, (13.0×12.5×13.0))	(41, 0.864)	(32, 0.563)
model11 (37, 0.789, (26.0×13.0×22.0))	(86, 0.894)	(66, 0.8)

Table 1: Models and their inflated variants. For each data set, (number of nodes, proximity score) is depicted.



(a) Model 1 (b) Model 2 (c) Model 3 (d) Model 4 (e) Model 5 (f) Model 6 (g) Model 7

(h) Model 8 (i) Model 9 (j) Model 10 (k) Model 11

Figure 4: Models used for the evaluation.

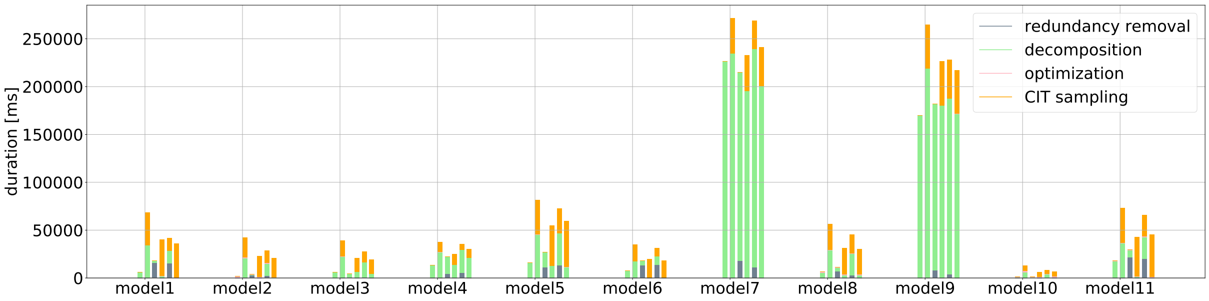


Figure 5: Timings for different pipeline configurations using data set 1. For each model configurations from left to right: $(0,0,0)$, $(1,0,0)$, $(0,0,1)$, $(0,1,1)$, $(1,0,1)$ and $(1,1,1)$.

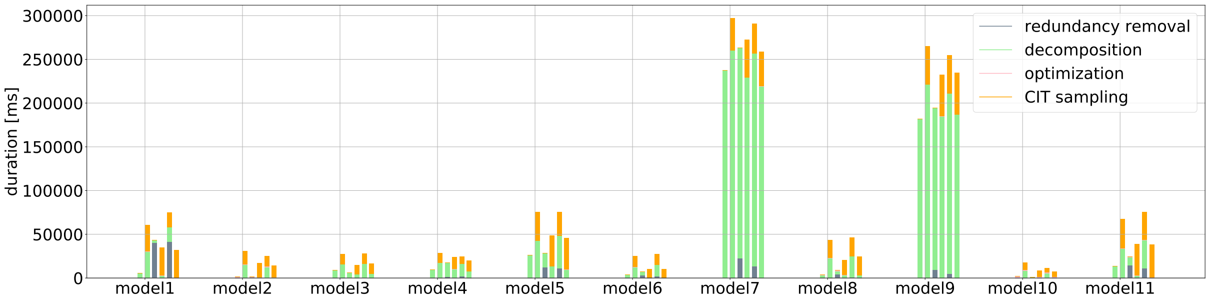
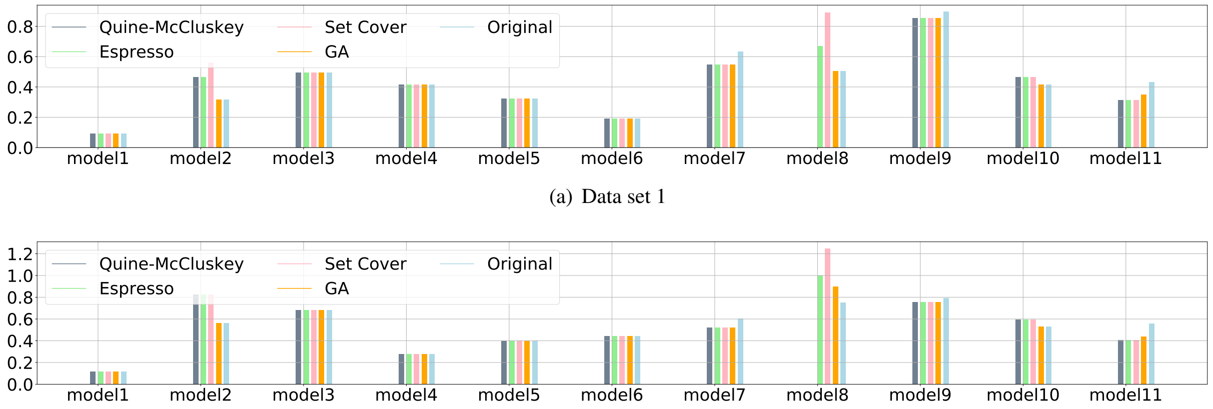


Figure 6: Timings for different pipeline configurations using data set 2. For each model configurations from left to right: $(0,0,0)$, $(1,0,0)$, $(0,0,1)$, $(0,1,1)$, $(1,0,1)$ and $(1,1,1)$.

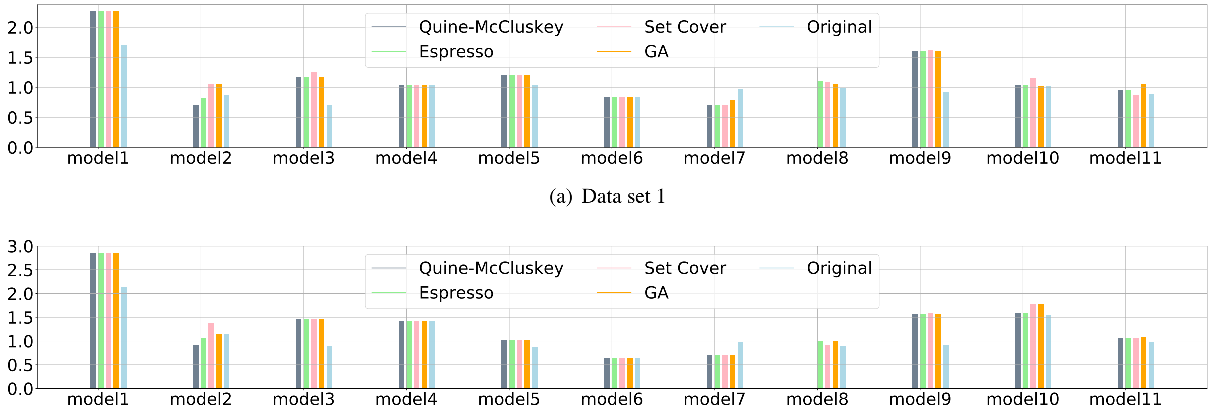
[BC04] Suzanne F Buchele and Richard H Crawford. Three-dimensional halfspace constructive solid geometry tree construction from implicit boundary representations. *Computer-Aided Design*, 36(11):1063–1073, 2004.



(a) Data set 1

(b) Data set 2

Figure 7: Ratio of input and output tree size for both data sets. The light blue bar 'Original' indicates the size of the initial hand-crafted expression.



(a) Data set 1

(b) Data set 2

Figure 8: Ratio of input and output tree proximity for both data sets. The light blue bar 'Original' indicates the proximity of the initial hand-crafted expression.

Models	Q-McC	Espresso	Set Cover	GA
model2	1572	1066	17305	21526
model8	-	1163	111017	79829
model10	3928	940	3347	22094
model11	3879	900	8594	90971

Table 2: RSO timings for data set 1 in [ms].

[BHMSV84] Robert K Brayton, Gary D Hachtel, Curt McMullen, and Alberto Sangiovanni-Vincentelli. *Logic minimization algorithms for VLSI synthesis*, volume 2. Springer Science & Business Media, 1984.

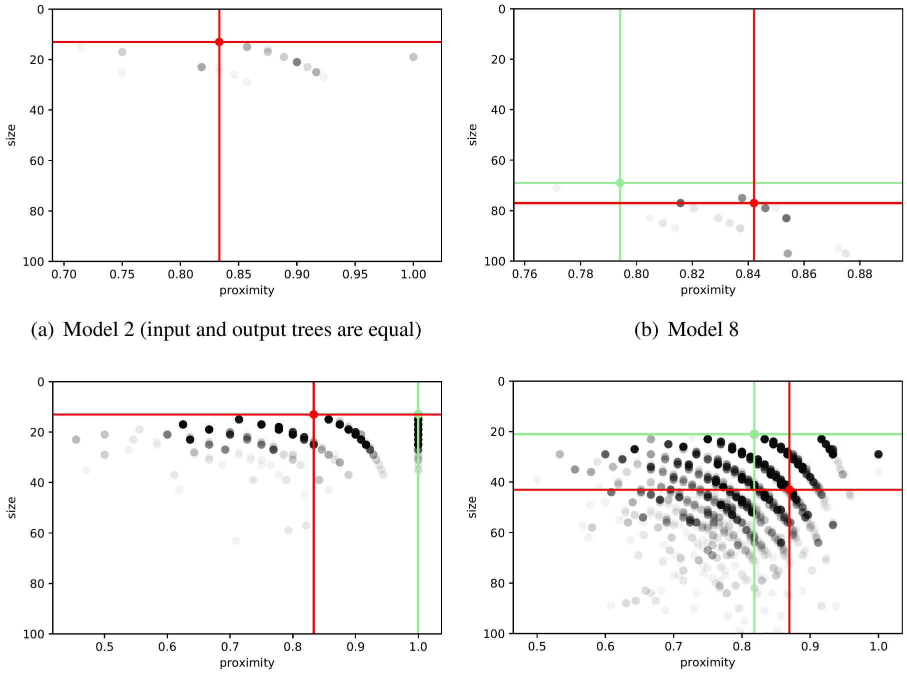
[DIP^{+}18] Tao Du, Jeevana Priya Inala, Yewen Pu, Andrew Spielberg, Adriana Schulz, Daniela Rus, Armando Solar-Lezama, and Wojciech Matusik. Inversecsg: Automatic conversion of 3d models to csg trees. In *SIGGRAPH Asia 2018 Technical Papers*, SIGGRAPH Asia '18, pages 213:1–213:16, New York, NY, USA, 2018. ACM.

[FFGLP19] Markus Friedrich, Pierre-Alain Fayolle, Thomas Gabor, and Claudia Linnhoff-Popien. Optimizing evolutionary csg tree extraction. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO 2019, pages 1183–1191. Association for Computing Machinery, 2019.

[FP16] Pierre-Alain Fayolle and Alexander Pasko. An evolutionary approach to the extraction of object construction trees from 3d point clouds. *Computer-Aided Design*, 74:1–17, 2016.

[Kar72] Richard M Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972.

[Luc14] Andrew Lucas. Ising formulations of many NP problems. *Frontiers in Physics*, 2:5, 2014.



(a) Model 2 (input and output trees are equal)

(b) Model 8

(c) Model 10

(d) Model 11

Figure 9: All trees representing the remaining solid produced by the GA for data set 2 with a geometric score of 1.0. The darker the point, the more trees have exactly this size/proximity combination. The red dot indicates the input tree that represents the remaining solid, the green one the selected resulting tree.

[MJ56] Edward J McCluskey Jr. Minimization of boolean functions. *Bell system technical Journal*, 35(6):1417–1444, 1956.

[O'R82] Joseph O'Rourke. Polygon decomposition and switching function minimization. *Computer Graphics and Image Processing*, 18(4):382 – 391, 1982.

[PASS95] Alexander Pasko, Valery Adzhiev, Alexei Sourin, and Vladimir Savchenko. Function representation in geometric modeling: concepts, implementation and applications. *The Visual Computer*, 11(8):429–446, 1995.

[Qui52] W. V. Quine. The problem of simplifying truth functions. *American Math. Monthly*, 59(8):521–531, 1952.

[Req80] Aristides G. Requicha. Representations for rigid solids: Theory, methods, and systems. *ACM Comput. Surv.*, 12(4):437–464, December 1980.

[Ric73] A. Ricci. A constructive geometry for computer graphics. *The Computer Journal*, 16(2):157–160, 1973.

[Ros11] Jarek Rossignac. Ordered boolean list (OBL): Reducing the footprint for evaluating boolean expressions. *IEEE Transactions on Visualization and Computer Graphics*, 17(9):1337–1351, 2011.

[Sha99] Vadim Shapiro. Well-formed set representations of solids. *International Journal of Computational Geometry & Applications*, 9(02):125–150, 1999.

[Sha07] Vadim Shapiro. Semi-analytic geometry with r-functions. *ACTA numerica*, 16:239–303, 2007.

[SV91a] Vadim Shapiro and Donald L. Vossler. Construction and optimization of CSG representations. *Computer-Aided Design*, 23(1):4–20, 1991.

[SV91b] Vadim Shapiro and Donald L Vossler. Efficient CSG representations of two-dimensional solids. *Journal of Mechanical Design*, 113(3):292–305, 1991.

[SV93] Vadim Shapiro and Donald L Vossler. Separation for boundary to CSG conversion. *ACM Transactions on Graphics (TOG)*, 12(1):35–55, 1993.

[Til84] Robert B. Tilove. A null-object detection algorithm for constructive solid geometry. *Commun. ACM*, 27(7):684–694, July 1984.

[WXW18] Qiaoyun Wu, Kai Xu, and Jun Wang. Constructing 3d CSG models from 3d raw point clouds. *Computer Graphics Forum*, 37(5):221–232, 2018.

Optimization of Degraded Document Image Binarization Method Based on Background Estimation

Hubert Michalak
West Pomeranian University
of Technology in Szczecin
Aleja Piastów 17
70-310 Szczecin, Poland
michalak.hubert@zut.edu.pl

Krzysztof Okarma
West Pomeranian University
of Technology in Szczecin
Aleja Piastów 17
70-310 Szczecin, Poland
okarma@zut.edu.pl

ABSTRACT

Binarization of document images is one of the most relevant pre-processing operations, leading to a significant decrease of the amount of information used during their further analysis. Since many document images, particularly historical, may be degraded over time, the application of some simple global thresholding methods usually lead to highly unsatisfactory results. A similar situation may occur for unevenly illuminated images, limiting the visibility of various shapes, representing not only the alphanumerical characters. A typical solution of this problem is the application of some adaptive thresholding methods, as well as more sophisticated solutions, proposed recently e.g. during Document Image Binarization Competitions (DIBCO) or TQ-DIB 2019 competition. Nevertheless, due to their relatively high computational demands, there is still a need of some faster methods, leading to high binarization accuracy for challenging benchmark datasets, such as DIBCO or Nabuco. Hence, the adaptation and optimization of the parameters of the fast thresholding method utilizing background estimation, proposed originally for the OCR purposes and verified for unevenly illuminated printed documents, is presented in this paper. The proposed solution has been optimized and verified using the state-of-the-art datasets containing 166 degraded document images together with their ground-truth binary equivalents, leading to better results, also in comparison to much slower adaptive thresholding methods. The performance of all methods used in comparisons has been determined using commonly accepted metrics, such as F-Measure, Accuracy, Distance Reciprocal Distortion (DRD) or Misclassification Penalty Metric (MPM), and relative execution time, calculated for all used image datasets.

Keywords

Image thresholding, document images, background estimation, binarization

1 INTRODUCTION

Document image binarization is one of the most challenging problems related to various applications of a fast shape analysis and recognition. Although many relatively simple well-known image thresholding methods may be successfully applied in some typical applications, such as Optical Character Recognition (OCR) using well illuminated scanned documents or shape classification of uniformly lightened objects, their usefulness for lower quality images is often significantly limited. In some industrial applications, such as video inspection, dedicated machine vision illuminators may be applied to ensure the uniformity of lighting conditions. On the other hand, assuming the presence of image distortions, affecting the background and character readability in document images, more sophisticated adaptive methods are used, which usually require the analysis of the neighbourhood of each individual pixel to determine the local threshold. In some applications, where the analysis of the document's structure, e.g. text line segmentation, particularly in non-Latin languages [Has19], is important, the appropriate binarization might be a critical step as well. A similar influence may also be observed in some modern solutions based on the application of deep Convolutional Neural Networks (CNN) for various tasks related to image and video analysis [Lef19].

Comparing the influence of non-uniform illumination with distortions typical for degraded historical document images, there is no doubt that the latter ones are more challenging. Hence, newly developed image binarization methods, proposed during recent years, are typically verified using the datasets used in Document Image Binarization Competitions (DIBCO) [Pra17]. Considering the need of balancing the processing speed and

https://doi.org/10.24132/CSRN.2020.3001.11

obtained accuracy, recently some other datasets have also been proposed, namely Nabuco and LiveMemory, used in Time-Quality Document Binarization Competition organized during the IAPR International Conference on Document Analysis and Recognition – ICDAR 2019 [Lin19]. Since many image binarization methods may be useful in some other applications, e.g. related to automation and robotics, where some devices with low computing performance may be applied, the results of binarization are typically assessed in a general way, using commonly accepted classification metrics usually based on the calculation of the numbers of properly and improperly classified foreground and background pixels.

Considering this, filling the gap between the fast global (e.g. well-known Otsu [Ots79] or Kittler [Kit86] methods) and relatively slow adaptive thresholding algorithms (e.g. proposed by Niblack [Nib86] or Sauvola [Sau00]) is an interesting alternative, also for document image binarization. The approach presented in this paper is based on the estimation of the image background, which may be initially removed to make the final binarization easier. Since the simple background subtraction without additional operations does not lead to satisfactory results, the optimization of some parameters of the proposed method using the state-of-the-art benchmarking datasets is presented, as well as some modifications of the method [Mic19a], initially used for the pre-processing of unevenly illuminated photographs.

The rest of the paper is organized as follows: in Section 2 a short overview of the most popular image binarization methods is provided, whereas Section 3 contains the description of the proposed approach with experimental results provided in Section 4 and discussed in Section 5. Finally, Section 6 concludes the paper.

2 OVERVIEW OF POPULAR IMAGE BINARIZATION METHODS

2.1 Global and Adaptive Thresholding

The simplest approach to image binarization is the choice of a single threshold, which is applied for all image pixels. Such global thresholding differ by a strategy of choosing the most appropriate threshold. Apart from the use of a fixed threshold (e.g. 50% of the brightness range), the most popular approach, belonging to the family of histogram based methods, is the minimization of the sum of inter-class variances, equivalent to maximization of inter-class variance, proposed by Otsu [Ots79]. A similar idea applied for histogram's entropy replacing the variances has been presented by Kapur *et al.* [Kap85].

Both these approaches work well only for bi-modal histograms assuming uniform lighting conditions, hence a different method of unimodal thresholding has been proposed by Rosin [Ros01], who suggested the use of a simple minimization of the distance between the straight line connecting the peak and the high end of the histogram to determine the threshold. A modified version of such approach, known as T-point method [Cou10], utilizes piecewise linear regression of the descending slope of the histogram, although its usefulness is limited to specific images containing small objects, e.g. electron microscope images, so it is unsuitable for typical document images.

Kittler and Illingworth [Kit86] proposed a global binarization method assuming that the grey levels are normally distributed and may be described by a mixture of two Gaussian functions. The criterion derived in Kittler's method minimizes the classification error probability dividing the histogram into two parts.

A region-based method based on the separate calculation of Otsu's thresholds for image regions as well as some additional features has been proposed by Chou *et al.* [Cho10]. Their values are further used to construct the adaptive decision rules using Support Vector Machines (SVM) assuming the 3×3 pixels windows. Nevertheless, this method has been dedicated for typical document images, relatively well illuminated, and assessed considering the OCR performance rather than using classification metrics for individual pixels.

An adaptive version of Otsu's method, known as *AdOtsu*, has been proposed by Moghaddam and Cheriet [Mog12], where a multi-layer classifier has been applied in combination with grid-based modelling. Although in this method the background modelling is used, starting from rough double-scale Sauvola's binarization, the application of the multi-scale framework and further post-processing steps additionally increases the computational effort. Nevertheless, a simpler and more popular adaptive method has been proposed by Wayne Niblack [Nib86], who has proposed the use of the difference of the local average brightness, calculated for the 3×3 pixels sliding window, and its scaled standard deviation (with default multiplier $k = 0.2$) as the local threshold.

Since the results of this method are often noisy, in its modified version proposed by Sauvola and Pietikäinen [Sau00], the dynamic range of standard deviation has been included into Niblack's formula. This approach has been further improved utilizing integral images to speed up the computations [Sha08], leading to ca. 20-fold increase of computational performance. The description of the multi-scale version of Sauvola's method, useful for both small and large objects inside a single document, may be found in Lazzara's paper [Laz14], where the robustness to font size has been the priority, regardless of relatively weak performance for historical documents.

Some other adaptive methods, being the modifications of Niblack's approach, have been proposed by various researchers, leading to some improvements of obtained results. A fast implementation of this method has been presented by Samorodova and Samorodov [Sam16], which utilizes the integral image for the calculation of the local mean values and the second order integral image for the calculation of the local variance. One of the most interesting modifications of Niblack's method has been proposed by Wolf and Jolion [Wol04], where the decision is made in terms of contrast instead of greyscale values, combining the advantages of Sauvola's and Niblack's methods, although two passes are necessary increasing the computational complexity of the method.

Another extension presented by Feng *et al.* [Fen04] requires the pre-processing step based on median filtering and bilinear interpolation of threshold values, calculated only for the centres of the local windows, introducing the additional parameters defined for two local windows used during calculations. Relatively good results may be obtained using the method, known as NICK, proposed by Khurshid *et al.* [Khu09], which assumes the shifting down of the threshold to improve the binarization of brighter documents and the use of 19×19 pixels local windows. It is worth noticing that the computational complexity of this method is lower in comparison fo Wolf's and Feng's algorithms. A more detailed overview of various Niblack inspired methods may be found in some recent survey papers [Khu09, Sax19, Sul19].

One of the most popular adaptive binarization methods, utilizing the integral images to prevent long calculations, available e.g. in MATLAB environment as the *adaptthresh* function, has been proposed by Bradley and Roth [Bra07]. In this method the local threshold is set as the specified percentage of the average local brightness, although in MATLAB implementation the local median as well as Gaussian weighted mean may also be used. Another well-known method proposed by Bernsen [Ber86] is the calculation of the local threshold as the average of the maximum and minimum intensities inside the square local window, although pixels in low contrast regions are always classified as background. Even simpler approach is the direct use of the local mean as the threshold (known as the *meanthresh* method).

One of the recent approaches [Mic19b] is based on the application of the local entropy filter for image preprocessing, assuming the 19×19 pixels windows, followed by classical Otsu's thresholding applied for the entropy map. Then, a binary mask subjected to morphological dilation is used for background estimation and removal. The last step of the method is the binarization, which may be conducted using well-known thresholding methods. Another recent idea is the application of multi-layered stack of regions [Mic19c] for the OCR purposes assuming the non-uniform illumination of document images, where the local threshold is calculated as the average of thresholds obtained for the regions "covering" the pixel using a formula based on the modified average intensity in the image region.

Considering that many methods proposed in recent years are computationally demanding, the paper is focused on relatively fast methods, possible to implement efficiently e.g. in mobile devices or embedded systems. Hence, a detailed description of some specialized methods, e.g. based on deep learning [Ten17, Vo18] is skipped in this paper.

2.2 Performance Evaluation

Each of newly proposed image binarization method should be verified experimentally using widely accepted benchmarking datasets as well as appropriate evaluation methods [Nti13]. The most typical performance metrics used for evaluation of binarization methods come from machine learning and classification assessment and are based on the calculation of the number of pixels classified properly as foreground (true positives – TP) or background (true negatives – TN). Incorrectly classified pixels are denoted as false positives (FP) or false negatives (FN), respectively. For document images white pixels, usually represented by "ones" in binary images, are considered as background (negatives) and therefore, an additional image negation is often applied before the calculations. Since all such metrics are based on the comparison of resulting images with "ground truth" (GT), such binary images have to be provided in each dataset.

The most typically used evaluation metrics are:

- precision – defined as the ratio of true positives to all positives: $PR = \frac{TP}{TP+TN}$,
- sensitivity / recall – defined as the ratio of true positives to condition positives (sum of true positives and false negatives): $RC = \frac{TP}{TP+FN}$,
- specificity / selectivity – defined as the ratio of true negatives to condition negatives (sum of false positives and true negatives): $SPEC = \frac{TN}{FP+TN}$,
- accuracy – defined as the sum of true positives and true negatives divided by all samples (pixels): $ACC = \frac{TP+TN}{TP+TN+FP+FN}$,
- F-Measure / F1-score – defined as the harmonic mean of precision and recall: $FM = 2 \cdot \frac{PR \cdot RC}{PR+RC}$, usually multiplied by 100%.

Using the above values derived from the confusion matrix, some additional metrics, such as e.g. balanced accuracy (used for imbalanced data) or Matthews correlation coefficient (MCC), may be calculated, similarly as Receiver Operating Characteristics (ROC) graphs. Nevertheless, these metrics mostly lead to the same conclusions, and therefore only the results of the most relevant ones have been presented in the paper (namely F-Measure and accuracy).

In addition to the above metrics, two other approaches for performance evaluation of image binarization methods have been proposed, which do not utilize typical classification statistics based only on the number of misclassified pixels. The first one, known as Distance Reciprocal Distortion (DRD) [Lu04], has been proposed as an alternative for the Peak Signal to Noise Ratio (PSNR) and the other metrics, which do not match well with subjective quality assessment. This metric considers also the mutual relations between pixels as the perception of distortions in binary images is different than for natural images. Hence, the weighting matrix has been proposed to consider the distances between the flipped pixels from the central pixel, not only the number of the misclassified pixels. Another useful tool is the Misclassification Penalty Metric (MPM) [You05], where the flipped pixels are penalized by their distances from the objects. To calculate the MPM values the GT object's borders should be extracted followed by distance calculation for misclassified pixels (FN and FP). The final MPM score is calculated as:

$$MPM = \frac{1}{2 \cdot D} \times \sum_{i,j} \left(d_{FN}^{i} + d_{FP}^{j} \right) \qquad (1)$$

where d_{FN}^{i} and d_{FP}^{j} are the distances of the i-th false negative and j-th false positive, respectively. To make the results more independent on image size, the normalization factor D has also been used, defined as the sum of the distances of all pixels to contours in the GT image. Nevertheless, although both these metrics are suitable for document images, they are not useful for halftone images (with dithering).

3 PROPOSED SOLUTION

Since the idea of the investigated approach is to have possibly low computational effort, preserving the performance comparable to adaptive algorithms, a relatively fast pre-processing step is used, which would not cause a significant increase of the computation time in contrast to e.g. median filtering. Hence, the global thresholding is assumed as the second stage, which should be conducted using an image with partially eliminated background information. Nevertheless, it has been assumed that some minor distortions may not be eliminated during the pre-processing stage, which should be suppressed by the final thresholding.

The initial background estimation is obtained using the image downsampling and further upsampling using one of typically used kernels: bilinear, bicubic and Lanczos (in two versions with *sinc* function of α equal to 2 and 3), as well as using simple nearest neighbour approach. For this purpose MATLAB's *imresize* function may be used with appropriate parameters, assuming the use of the same kernel for both operations. The effects obtained after the use of consecutive steps of the algorithm are shown on the top images of Fig. 1, where the influence of image resampling may be observed as well as the extracted details representing the foreground text, used in further steps. A crucial parameter, subjected to optimization, is the kernel size *sc* (referred also as the scale factor), significantly influencing the obtained results as shown in four plots in Fig. 2.

The next steps are the subtraction of two images to obtain the image with partially suppressed background, which is further enhanced by the increase of contrast. To maintain the speed of the method a simple multiplication by the contrast correction factor k (with saturation of bright pixels) has been used, followed by image negation. Obviously, these two steps may be reversed, assuming the saturation of dark pixels for the increase of image contrast after negation. Such obtained image, using the optimized contrast correction factor is then subjected to final binarization, which may be conducted using a fixed threshold or any global thresholding method, e.g. Otsu's binarization, which has led to the best results during the experiments. The results of these three steps for a sample document image are illustrated in the bottom part of Fig. 1.

During the experiments, various interpolation kernels used during down- and upsampling, as well as different final thresholding methods, have been checked. Surprisingly, the best results have been achieved using the bilinear kernel combined with Otsu's thresholding in the final step. The illustration of the results of the metrics obtained during the optimization of the parameters is presented in Figures 2 and 3. Analysing these plots, the combination of the smallest reasonable contrast correction factor with the possibly universal scale factor, leading to good results for the datasets containing images of different sizes, should be chosen (in this case $k = 0.5$ for contrast and $sc = 32$, respectively).

4 EXPERIMENTAL RESULTS

To verify the usefulness of the proposed optimized method, several known algorithms, shortly presented in Section 2.1, have been used and the comparisons of the accuracy, F-Measure, DRD and MPM metrics have been made using four datasets. During the experiments the 116 images from DIBCO 2009 – 2018 datasets have been aggregated into a single dataset and the three remaining ones have been proposed in conjunction with the Time-Quality Document Binarization Competition (TQ-DIB). Two of them contain the images of letters and postcards written and typed by Brazilian writer and

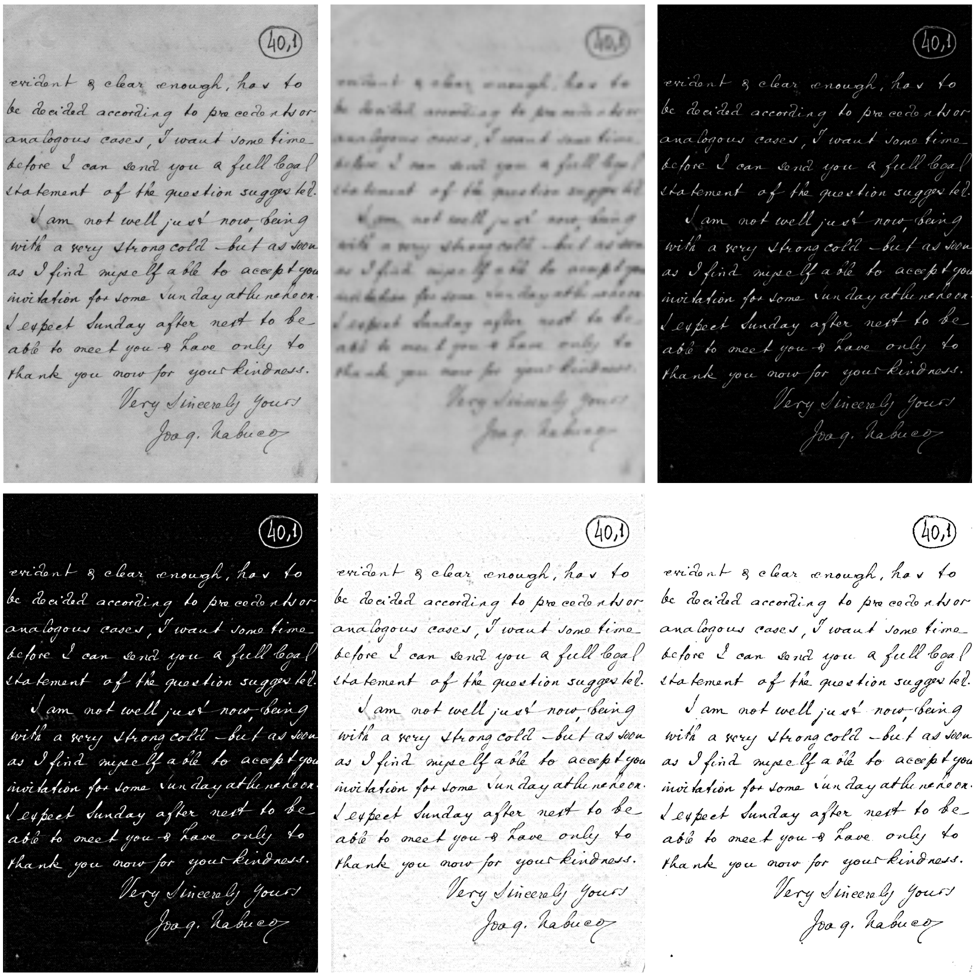


Figure 1: Results obtained after consecutive steps of the proposed method for a sample document image from Nabuco dataset – from left to right: original image, estimated background and their difference (top row), increased contrast, negation, final result after Otsu's thresholding (bottom row).

statesman Joaquim Nabuco – one of them, containing 15 images, is referred as "Nabuco dataset" whereas the second one consisting of 20 images, used in TQ-DIB 2019 competition [Lin19], is referred as "Nabuco competition". However, most of the images in those two datasets are different, hence they have been considered separately during the experiments. The fourth dataset, called LiveMemory, available at the website of the TQ-DIB organizers as well[1], contains 15 document images selected from the proceedings of the Brazilian Telecommunications Society. Obviously, all the datasets contain the respective binary ground-truth images.

Regardless of the performance metrics, the approximate execution times have also been determined using the same PC with Intel Core-i7 CPU and 16GB of RAM with MATLAB 2018b running on 64-bit Windows 10 operating system. Although the presented times should be considered as approximate, it is possible to compare the computational effort of individual algorithms, even though the implementation of the proposed method has not been optimized in terms of processing speed. As expected, the proposed method has turned out to be obviously slower than global thresholding but still significantly faster than all adaptive methods.

To make the comparison of the execution time more universal and hardware independent all execution times have been determined relatively to the global Otsu's method. The advantages of some adaptive methods utilising integral images may also be easily observed, e.g. comparing time rank positions of both Bradley methods. Nevertheless, it is worth to note that the proposed method is faster even in comparison with the global Kittler method.

Some more detailed results of the four considered performance metrics for individual datasets are presented in Tables 1 and 2, demonstrating the advantages of the proposed approach. For a reliable comparison of all methods both in terms of time and quality, they have been ranked according to each of four performance

[1] https://dib.cin.ufpe.br/

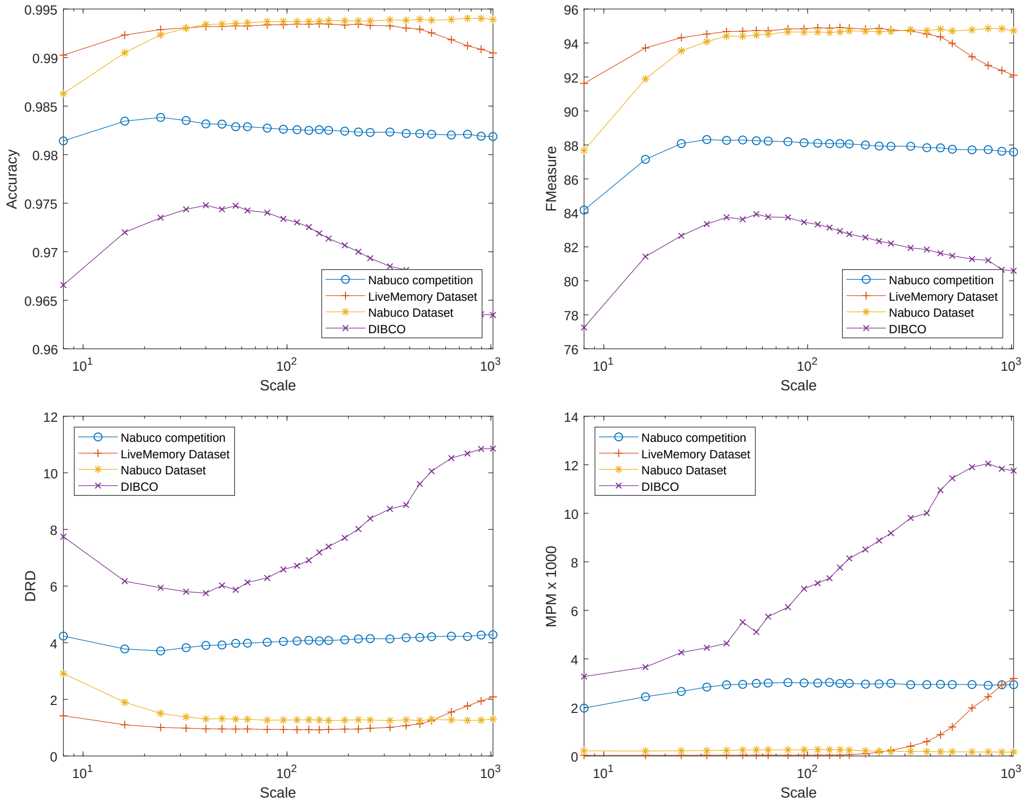


Figure 2: Illustration of results obtained during the optimization of parameter *sc*.

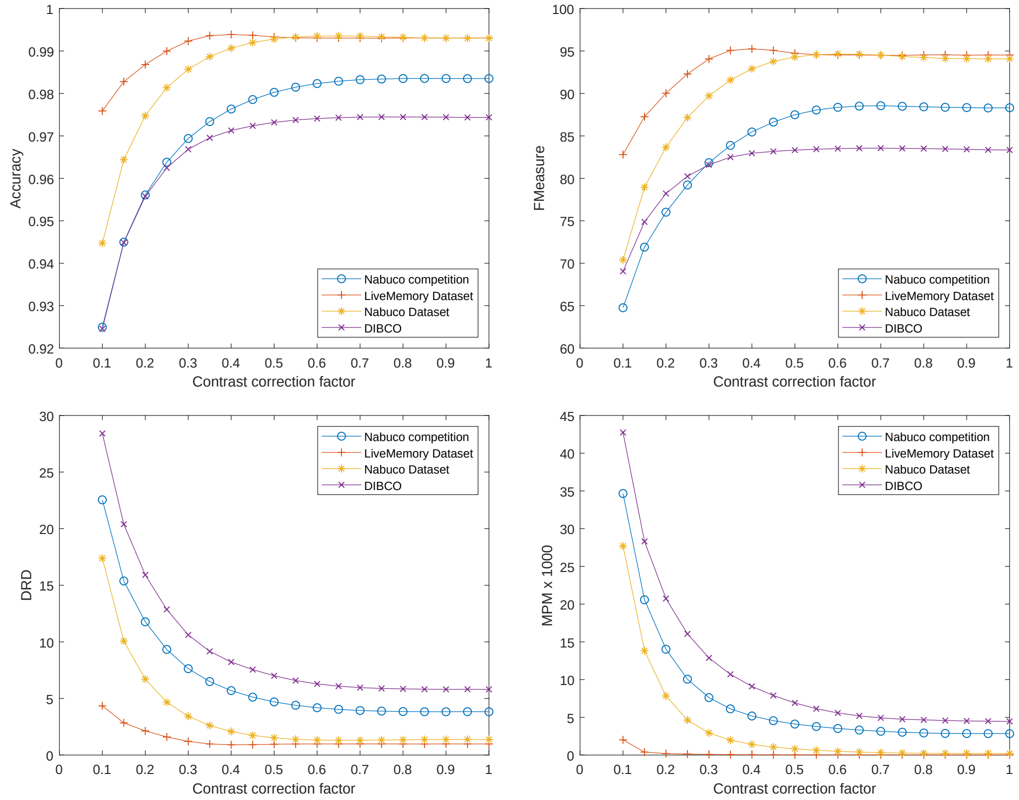


Figure 3: Illustration of results obtained during the optimization of parameter *k*.

Binarization method	Reference	DIBCO				LiveMemory			
		FM	ACC	DRD	MPM ×1000	FM	ACC	DRD	MPM ×1000
Otsu	[Ots79]	78.77	0.948	16.72	22.22	89.64	0.986	3.78	6.65
Kittler	[Kit86]	70.96	0.920	28.24	32.94	70.27	0.949	11.19	16.16
Meanthresh	–	81.47	0.969	9.01	10.11	90.47	0.987	2.04	0.23
Niblack	[Nib86]	74.68	0.951	16.29	22.73	90.25	0.988	2.32	1.26
Sauvola	[Sau00]	78.08	0.957	12.11	12.13	88.97	0.985	2.43	0.17
Wolf	[Wol04]	76.24	0.953	13.43	13.27	86.46	0.981	3.08	0.22
Feng	[Fen04]	66.76	0.933	22.54	22.86	90.99	0.989	2.06	0.29
Bradley (Mean)	[Bra07]	77.43	0.954	16.07	22.20	88.66	0.985	2.48	0.20
Bradley (Gaussian)	[Bra07]	75.33	0.946	18.99	24.97	87.36	0.982	2.86	0.57
NICK	[Khu09]	69.90	0.960	9.21	**3.50**	89.22	0.989	1.68	**0.03**
Chou	[Cho10]	74.02	0.940	24.77	37.84	86.07	0.980	5.89	18.50
Region-based (1 layer)	[Mic19c]	76.29	0.956	13.30	14.89	91.36	0.989	1.82	0.17
Region-based (2 layers)	[Mic19c]	78.68	0.962	11.68	13.71	92.63	0.990	1.55	0.14
Region-based (4 layers)	[Mic19c]	79.34	0.963	11.25	13.07	92.90	0.991	1.49	0.13
Region-based (6 layers)	[Mic19c]	79.44	0.964	11.18	12.93	92.94	0.991	1.48	0.13
Region-based (8 layers)	[Mic19c]	79.51	0.964	11.13	12.82	92.96	0.991	1.48	0.13
Region-based (12 layers)	[Mic19c]	79.53	0.964	11.11	12.75	92.97	0.991	1.48	0.13
Region-based (16 layers)	[Mic19c]	79.55	0.964	11.09	12.73	92.98	0.991	1.48	0.13
Entropy filter + Otsu	[Mic19b]	80.73	0.966	11.22	10.79	94.43	0.993	1.18	0.09
Proposed (bilinear; k=0.5; sc=32)	[Mic19a]	**83.33**	**0.973**	**7.00**	6.91	**94.74**	**0.993**	**0.96**	0.04

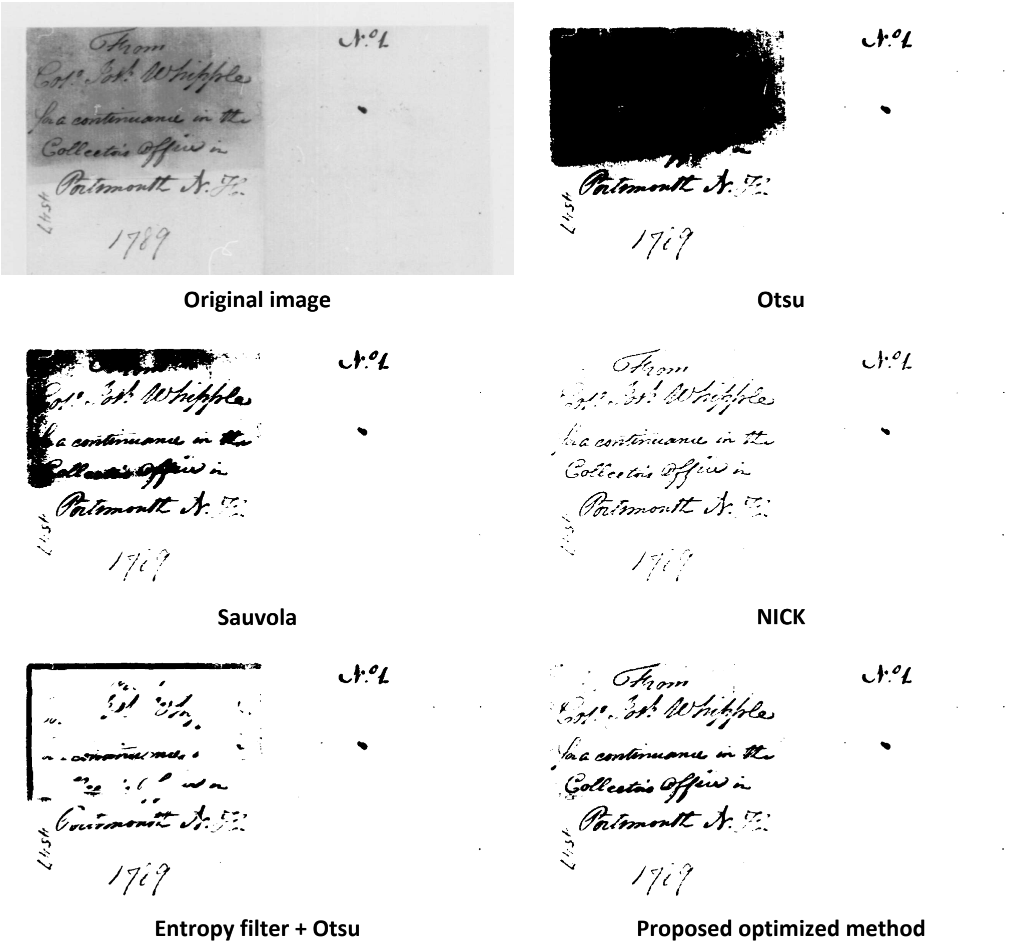
Table 1: Results of performance metrics obtained for DIBCO and LiveMemory datasets

Binarization method	Reference	Nabuco competition (TQ-DIB)				Nabuco dataset			
		FM	ACC	DRD	MPM ×1000	FM	ACC	DRD	MPM ×1000
Otsu	[Ots79]	86.29	0.978	5.36	4.15	94.01	0.993	1.63	0.37
Kittler	[Kit86]	73.22	0.950	13.24	9.34	78.71	0.967	9.01	3.40
Meanthresh	–	85.74	0.978	5.59	6.47	90.98	0.988	3.14	3.60
Niblack	[Nib86]	77.10	0.961	11.76	18.97	82.81	0.974	8.28	13.64
Sauvola	[Sau00]	75.25	0.957	11.09	8.00	83.25	0.975	6.27	2.39
Wolf	[Wol04]	76.24	0.959	10.54	7.41	83.86	0.976	6.04	1.83
Feng	[Fen04]	76.14	0.962	10.94	10.72	82.11	0.973	7.92	7.12
Bradley (Mean)	[Bra07]	74.80	0.953	12.27	11.96	80.97	0.971	7.92	7.25
Bradley (Gaussian)	[Bra07]	73.05	0.949	13.76	12.53	78.20	0.966	9.91	8.14
NICK	[Khu09]	84.58	**0.982**	**4.43**	**2.89**	89.90	0.988	2.58	0.71
Chou	[Cho10]	77.96	0.963	15.05	21.95	80.36	0.965	18.56	34.04
Region-based (1 layer)	[Mic19c]	78.64	0.965	9.78	12.25	85.13	0.978	6.09	6.96
Region-based (2 layers)	[Mic19c]	81.23	0.970	8.46	11.20	87.63	0.982	5.05	6.55
Region-based (4 layers)	[Mic19c]	81.96	0.971	8.10	10.50	88.39	0.983	4.72	6.15
Region-based (6 layers)	[Mic19c]	82.08	0.971	8.04	10.38	88.50	0.983	4.68	6.08
Region-based (8 layers)	[Mic19c]	82.14	0.971	8.00	10.30	88.55	0.984	4.65	6.02
Region-based (12 layers)	[Mic19c]	82.17	0.971	7.97	10.24	88.58	0.984	4.64	5.99
Region-based (16 layers)	[Mic19c]	82.19	0.971	7.96	10.22	88.59	0.984	4.63	5.98
Entropy filter + Otsu	[Mic19b]	87.24	0.980	4.78	3.70	**94.48**	**0.993**	**1.44**	**0.32**
Proposed (bilinear; k=0.5; sc=32)	[Mic19a]	**87.49**	0.980	4.70	4.11	94.29	**0.993**	1.52	0.82

Table 2: Results of performance metrics obtained for both Nabuco datasets

Binarization method	Relative execution time	Time rank	Performance (Quality) rank	Time-Quality rank
Otsu	1.00	**1**	6	2
Kittler	22.87	5	18	10
Meanthresh	37.37	6	2	3
Niblack	71.09	9	15	11
Sauvola	71.68	10	11	9
Wolf	76.14	11	14	13
Feng	197.89	14	17	19
Bradley (Mean)	18.09	4	15	5
Bradley (Gaussian)	172.63	13	20	20
NICK	65.86	8	4	4
Chou (without SVM)	6.72	2	18	6
Region-based (1 layer)	53.70	7	13	6
Region-based (2 layers)	112.78	12	12	11
Region-based (4 layers)	221.94	15	10	13
Region-based (6 layers)	342.16	16	9	13
Region-based (8 layers)	474.42	17	8	13
Region-based (12 layers)	713.10	19	7	18
Region-based (16 layers)	959.53	20	5	13
Entropy filter + Otsu	646.34	18	2	6
Proposed (bilinear; k=0.5; sc=32)	12.06	**3**	**1**	**1**

Table 3: Approximate aggregated execution times relative to Otsu's method and obtained ranking scores



Original image

Otsu

Sauvola

NICK

Entropy filter + Otsu

Proposed optimized method

Figure 4: Results of the use of selected binarization methods for a sample challenging image used in experiments.

metrics for 4 datasets, leading to 16 rankings. The final quality ranking has been obtained after calculating the sum of the obtained positions, whereas time-quality ranking has been obtained in the same manner using positions in the final quality and execution time ranking.

5 DISCUSSION OF RESULTS

Analysing the results presented in Tables 1 and 2, much worse results may be noticed for DIBCO dataset, caused by much higher number of images, as well as the presence of some more challenging images in the latest DIBCO subsets. The Nabuco dataset seems to be much less challenging, as relatively good performance for these images may be noticed even using classical Otsu's method. Regardless of the dataset and the metric, the proposed method is always in top three methods. Since for both Nabuco datasets slightly better results may be obtained using the entropy based method [Mic19b] or NICK thresholding [Khu09], an important advantage of the method based on background estimation is its lower computational complexity, as illustrated in Table 3, where achieved execution times in relation to Otsu's method have been presented together with ranking scores.

A comparison of the results obtained for a sample challenging image is presented in Fig. 4, where the advantages of the proposed method are clearly visible. Since the presented sample image contains different shadows combined with rapid changes of intensity at the edges, the use of global thresholding methods is inappropriate. Nevertheless, serious problems may be observed also using adaptive methods, such as Sauvola and NICK, which have performed much better than the other methods in this case, in contrast to the proposed approach. Hence, the usefulness of the binarization based on background estimation for degraded document images has been confirmed, also considering the processing time constraints.

Nevertheless, analysing the potential limitations of the proposed approach, the main challenge might be a combination of degradations typical for document images with highly non-uniform illumination, where some more sophisticated time-consuming methods may be necessary, especially for images captured by mobile devices in unknown lighting conditions.

6 CONCLUDING REMARKS

The proposed optimized binarization method based on the fast background estimation using the down- and upsampling with bilinear kernel combined with increase of contrast and Otsu's thresholding, makes it possible to achieve very good performance for state-of-the-art image binarization datasets. Since it combines a good performance with relatively short processing time, it may be an interesting alternative for classical algorithms, particularly for low computational power devices. As the practical usefulness of this approach has also been verified in terms of the OCR accuracy, it may be considered as a universal binarization method for low quality document images.

A similar performance may be obtained using another two-stage method proposed recently [Mic19b], utilizing entropy filtering as the pre-processing step, followed by Otsu's thresholding as well. Nevertheless, this method is much slower even in comparison to adaptive methods. A combination of both these preprocessing approaches may be however considered in future research, although such an approach would be time-consuming and potentially useful only for off-line document image analysis. Another direction of our further work may be the application of the proposed approach for challenging pre-processing of images of unevenly illuminated industrial nameplates, captured by cameras mounted in mobile devices, subjected to further text recognition.

7 REFERENCES

[Ber86] Bernsen, J. Dynamic thresholding of grey-level images, in *Proc. 8th International Conference on Pattern Recognition (ICPR)*, pp. 1251-1255, 1986.

[Bra07] Bradley, D., and Roth, G. Adaptive thresholding using the integral image. *Journal of Graphics Tools* 12(2):13-21, 2007.

[Cho10] Chou, C.-H., Lin, W.-H., and Chang F. A binarization method with learning-built rules for document images produced by cameras. *Pattern Recognition* 43(4):1518-1530, 2010.

[Cou10] Coudray, N., Buessler, J.-L., and Urban, J.-P. Robust threshold estimation for images with unimodal histograms. *Pattern Recognition Letters* 31(9):1010-1019, 2010.

[Fen04] Feng,, M.-L., and Tan, Y.-P. Adaptive binarization method for document image analysis, in *Proc. 2004 IEEE Internatonal Conference on Multimedia and Expo (ICME)*, vol. 1, pp. 339-342, 2004.

[Has19] Hassina, B. Title segmentation in Arabic document pages, in *Proc. 27. International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG)* – part II (*Computer Science Research Notes*, vol. 2902), pp. 45-50, 2019.

[Kap85] Kapur, J.N., Sahoo, P.K., and Wong, A.K.C. A new method for gray-level picture thresholding using the entropy of the histogram. *Computer Vision, Graphics, and Image Processing* 29(3):273-285, 1985.

[Khu09] Khurshid, K., Siddiqi, I., Faure, C., and Vincent, N. Comparison of Niblack inspired binarization methods for ancient documents, in *Proc. SPIE – Document Recognition and Retrieval XVI*, article no. 724709, 2009.

[Kit86] Kittler, J., and Illingworth, J. Minimum error thresholding. *Pattern Recognition* 19(1): 42-47, 1986.

[Laz14] Lazzara, G., and Geráud, T. Efficient multiscale Sauvola's binarization. *International Journal on Document Analysis and Recognition* 17:105-123, 2014.

[Lef19] Lefkovits, S., Lefkovits, L., Szilágyi, L. CNN approaches for dorsal hand vein based identification, in *Proc. 27. International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG)* – part II (*Computer Science Research Notes*, vol 2902), pp. 51-60, 2019.

[Lin19] Lins, R, Bernardino, R., and Jesus, D.M. A quality and time assessment of binarization algorithms, in *Proc. 15th IAPR International Conference on Document Analysis and Recognition (ICDAR)* pp. 1444-1450, 2019.

[Lu04] Lu, H., Kot, A.C, and Shi, Y.Q. Distance-reciprocal distortion measure for binary document images. *IEEE Signal Processing Letters* 11(2):228-231, 2004.

[Mic19a] Michalak, H., Okarma, K. Fast binarization of unevenly illuminated document images based on background estimation for optical character recognition purposes. *Journal of Universal Computer Science* 25(6):627-646, 2019.

[Mic19b] Michalak, H., Okarma, K. Improvement of image binarization methods using image preprocessing with local entropy filtering for alphanumerical character recognition purposes. *Entropy* 21(6), article no. 562, 2019.

[Mic19c] Michalak, H., Okarma, K. Adaptive Image binarization based on multi-layered stack of regions, in *Proc. of International Conference on Computer Analysis of Images and Patterns (CAIP)*, LNCS, vol. 11679, pp. 281-293, 2019.

[Mog12] Moghaddam, R.F, and Cheriet, M. AdOtsu: an adaptive and parameterless generalization of Otsu's method for document image binarization. *Pattern Recognition* 45(6):2419-2431, 2012.

[Nib86] Niblack W. An introduction to digital image processing. Prentice Hall, Englewood Cliffs, 1986.

[Nti13] Ntirogiannis, K., Gatos, B., and Pratikakis, I. Performance evaluation methodology for historical document image binarization. *IEEE Transactions on Image Processing* 22(2):595-609, 2013.

[Ots79] Otsu, N. A threshold selection method from gray-level histograms/ *IEEE Trans. on Systems, Man, and Cybernetics* 9(1):62-66, 1979.

[Pra17] Pratikakis, I., Zagoris, K., Barlas, G., and Gatos, B. ICDAR2017 competition on document image binarization (DIBCO 2017), in *Proc. 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, pp. 1395-1403, 2017.

[Ros01] Rosin, P. Unimodal thresholding. *Pattern Recognition* 34(11):2083-2096, 2001.

[Sam16] Samorodova, O.A., and Samorodov, A.V. Fast implementation of the Niblack binarization algorithm for microscope image segmentation. *Pattern Recognition and Image Analysis* 26(3):548-551, 2016.

[Sau00] Sauvola, J., and Pietikäinen, M. Adaptive document image binarization. *Pattern Recognition* 33(2):225-236, 2000.

[Sax19] Saxena, L.P. Niblack's binarization method and its modifications to real-time applications: a review. *Artificial Intelligence Review*, 51(4):673-705, 2019.

[Sha08] Shafait, F., Keysers, D., and Breuel, T.M. Efficient implementation of local adaptive thresholding techniques using integral images, in *Proc. SPIE – Document Recognition and Retrieval XV*, article no. 681510, 2008.

[Sul19] Sulaiman, A., Omar, Kh., and Nasrudin, M.F. Degraded historical document binarization: a review on issues, challenges, techniques, and future directions. *Journal of Imaging* 5(4), article no. 48, 2019.

[Ten17] Tensmeyer, C., and Martinez, T. Document image binarization with fully convolutional neural networks, in *Proc. of 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, pp. 99-104, 2017.

[Vo18] Vo, Q.N., Kim, S.H., Yang, K.J., and Lee, G. Binarization of degraded document images based on hierarchical deep supervised network. *Pattern Recognition* 74:568-586, 2018.

[Wol04] Wolf, C., and Jolion, J.-M. Extraction and recognition of artificial text in multimedia documents. *Formal Pattern Analysis & Applications* 6(4):309-326, 2004.

[You05] Young, D.P., and Ferryman, J.M. PETS metrics: on-line performance evaluation service, in *Proc. IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pp. 317-324, 2005.

Variable-Radius Offset Surface Approximation on the GPU

Ann-Christin Woerl[1]
anwoerl@students.uni-mainz.de

Elmar Schoemer[1]
schoemer@uni-mainz.de

Ulrich Schwanecke[2]
ulrich.schwanecke@hs-rm.de

[1] Institute of Computer Science, Johannes Gutenberg University Mainz, Germany
[2] Computer Vision and Mixed Reality Group, RheinMain University of Applied Sciences Wiesbaden Rüsselsheim, Germany

ABSTRACT

Variable-radius offset surfaces find applications in various fields, such as variable brush strokes in 2D and 3D sketching and geometric modeling tools. In forensic facial reconstruction the skin surface can be inferred from a given skull by computing a variable-radius offset surface of the skull surface. Thereby, the skull is represented as a two-manifold triangle mesh and the facial soft tissue thickness is specified for each vertex of the mesh. We present a method to interactively visualize the wanted skin surface by rendering the variable-radius offset surfaces of all triangles of the skull mesh. We have also developed a special shader program which is able to generate a discretized volumetric representation that can be transformed into a skin mesh. In addition, we show the usefulness of our method to calculate classical Minkowski sums and demonstrate its use for packing problems.

Keywords

Variable-radius offsetting, Minkowski sum, Shader based shape approximation

1 INTRODUCTION

Offset surfaces are an important tool in CAD/CAM applications. Here they are used for various tasks, such as the construction of blend surfaces or the consideration of the radius of a milling cutter. They can also be used to account for the shrinkage of thermoplastics during 3D printing.

While simple variable-radius blend surfaces such as rolling ball surfaces are a widely used tool in geometric modeling, offset surfaces with freely varying radii are less common. One typical use case for these general variable-radius offset surfaces is the modeling of brush-strokes in 2D and 3D painting and modeling applications. Another rather niche application of variable-radius offset surfaces is forensic facial reconstruction. This is about reconstructing the face of a person on the basis of a skull.

Usually forensic facial reconstruction is performed by experts who, based on a lot of experience and statistical knowledge, apply the skin thickness to various points on a particular skull. In the next step, the gaps between these points are filled, which leads to a three-dimensional reconstruction of the face. In [1] the authors present an automatic method for forensic facial reconstruction. Their approach is based on a parametric template skull and dense statistics of the facial soft tissue thickness (FSTT). In the first step of their forensic facial reconstruction method, the skull template is fitted to the given skull. Then, based on the FSTT, a variable-radius offset surface is constructed. Finally, a head-template is fitted to the variable-radius offset surface. The most time-consuming part of this process is the creation of the variable-radius offset surface. In this paper we present a new method for the efficient generation of such a surface.

An explicit exact formula for a variable-radius offset surface can be complicated and time-consuming to evaluate. However, in many applications there is no need for an exact representation. Thus, we opted for an approximate representation, which leads to a better runtime. Our approach is motivated by the following simple observation: The intersection of a straight line with a solid defined by its boundary surface can be described by a set of disjoint intervals. Thus, we compute the intersection of three sets of parallel lines of a Cartesian 3D grid with a given variable-radius offset surface to approximate the surface. This also makes it possible to generate a triangle mesh of the offset (and simultaneously the inset) surface using an adapted Marching Cubes algorithm. Our algorithm works incrementally by dividing the given triangular mesh into single triangles and calculating the intersection intervals of the straight lines and the variable-radius offset surface of

https://doi.org/10.24132/CSRN.2020.3001.12

each individual triangle. Due to this decomposition we obtain a set of intervals per line. Each interval is directly joined with previously calculated intervals. We use a GPU to perform these computations in parallel. Therefore, we must pay particular attention to avoid incoherent memory access while computing the union of a new interval with an existing sorted list of disjoint intervals. Furthermore, our method can be used to calculate classical Minkowski sums. Another advantage of the proposed representation is that it can be used to quickly check whether a point is part of the Minkowski sum, which can be used for collision detections.

The paper is structured as follows: In the next section, we give a brief overview of work dealing with variable-radius offset surfaces and the related topic of Minkowski sums. Next, we describe our proposed data structure for representing variable-radius offset surfaces such that they can be efficiently evaluated on a GPU and avoid incoherent memory access. In section four we present a detailed description of our shader-based algorithm for calculating variable-radius offset surfaces and Minkowski sums on a GPU. In the penultimate chapter we present two application examples. The paper finally ends with a short summary and some thoughts on further improvements of the presented algorithm.

2 RELATED WORK

Fixed-radius offset surfaces are well studied since they are often used in CAD/CAM modeling. In contrast, variable-radius offset surfaces are less well known. Qun and Rokne [16] developed explicit formulas for variable-radius offset surfaces of parametric surfaces. In [18], Thiery et al. describe a shape representation algorithm using variable-radius offset surfaces. They introduce so called sphere-meshes, which are a connected set of spheres that are linearly interpolated over a triangular area. Wang and Manocha [21] describe an algorithm for fixed-radius offset computation on the GPU. Based on layered depth images, they calculate structured points located on a 3D grid, for which they compute the union of many spheres centered at these points. They use spatial hashing to find intersections. Gietzen et al. [10] use dense statistics of facial soft tissue thickness to reconstruct facial structures by a union of spheres. Achenbach et al. [1] further improved this approach by replacing the discontinuous union of spheres with a global optimized sphere-mesh, which is defined as the zero-set of a signed distance function. From this implicit representation, an explicit representation of the surface mesh can be obtained by using the marching cubes algorithm [15]. We present an algorithm that is able to interactively render a variable-radius offset surface based on per pixel calculations. Our method provides a higher accuracy as the one proposed in [1], while being much less time-consuming. In contrast to [21], our algorithm is able to work directly with 3D meshes without making an intermediate step via layered depth images and using hash functions for finding intersecting rays. Furthermore it can be easily generalized to calculate the Minkowski sum of polyhedra.

In general, there are two basic approaches to determine the Minkowski sum of two polyhedra. The first approach divides the polyhedra into convex components. Then the Minkowski sum is calculated for all pairwise convex components. Afterwards, the union of the individual results must be formed. Hachenberger [11] uses this approach to develop an algorithm for calculating the Minkowski sum of two non-convex polyhedra with exact geometric predicates, which, however, leads to high computation times. Varadhan and Manocha [20] also determine the Minkowski sum by convex decomposition of the polyhedra. They were limited to an approximate representation using a voxel grid representing a signed distance field, resulting in a much faster algorithm.

The second approach for calculating Minkowski sums for non-convex polyhedra is to determine the convolution of two polyhedra, which is a superset of the Minkowski sum. Afterwards, redundant surfaces must be removed by clever trimming. Lien [14] uses this approach and performs trimming in 2D using collision detection. This can be done efficiently by accepting a loss of accuracy. Campen and Kobbelt [3] use different criteria for trimming but are limited to the outer boundary of the Minkowski sum. This leads to a restricted field of applications due to the fact that inner boundaries are common. Li and McMains [13] describe an approach to represent the outer boundary of the Minkowski sum using voxelization on the GPU. The accuracy of their approximative approach is limited by the resolution of the voxel grid. In addition, no inner surfaces can be represented here either. Kyung et al. [12] describe a robust convolutional algorithm to calculate the Minkowski sum of two polyhedra, that finds and removes intersecting facets using kd-trees and achieves high accuracy.

3 SHADER BASED COMPUTATION

In our approach, the variable-radius offset surface is represented by three sets of parallel lines of a Cartesian 3D grid. This results in three independent calculations, each of the three orthogonal surfaces of the Cartesian 3D grid being considered a 2D grid. Thus, the image plane is discretized with a fixed grid width, with the grid points corresponding to the pixels. At each pixel our algorithm determines the intersection of a straight line with the given variable-radius offset surface. This results in a set of disjoint intervals for each pixel. Knowing the world coordinates of the underlying grid, it is possible to check in quasi constant time whether any position is part of the variable-radius offset surface or not. Based on the distance between the origin of each line and the variable-radius offset surface, the corresponding index in the array can be determined and the associated intervals can be found.

We use graphics hardware to compute the variable-radius offset surface for arbitrary triangle meshes. Since each object can be represented approximately by a 3D triangle mesh, this is not a limitation of our algorithm. Furthermore, this approach enables our algorithm to calculate not only the offset surface but also the inset surface simultaneously. A triangle mesh consists of three-dimensional vertex data with additional connectivity information. The vertices may also contain some additional properties such as normals or colors. As the triangle mesh passes through the rendering pipeline, the shader receives variable-radius data for each vertex of the mesh. All triangles of the mesh are considered independently of each other. In the following, we start by describing the general procedure for calculating the variable-radius offset surface for a single triangle which we will extend to multiple triangles afterwards. Implementations of the essential parts of the geometry and fragment shader can be found in Appendix A.

Vertex processing takes place first. Here, each individual vertex is processed by the vertex shader. The vertices are transformed into the two-dimensional image plane by multiplying them with the modelview and projection matrix.

The vertex shader is followed by a geometry shader. As an input it receives a single primitive and generates zero or more primitives as an output. We use one triangle as input and create two triangles as output. These two triangles are constructed as follows. Let $a, b, c \in \mathbb{R}^3$ be the vertices of the input triangle T_{abc} projected along the z-axis and

$$\begin{aligned} \begin{pmatrix} x_{min} \\ y_{min} \end{pmatrix} &= \begin{pmatrix} \min\{a_x, b_x, c_x\} \\ \min\{a_y, b_y, c_y\} \end{pmatrix}, \\ \begin{pmatrix} x_{max} \\ y_{max} \end{pmatrix} &= \begin{pmatrix} \max\{a_x, b_x, c_x\} \\ \max\{a_y, b_y, c_y\} \end{pmatrix} \end{aligned}$$

the lower left and upper right point of its axis-aligned bounding box. Furthermore, let $r_a, r_b, r_c \in \mathbb{R}$ be the variable-radii at the vertices and $r = \max\{r_a, r_b, r_c\}$ the maximum of these. A superset of the projection of the variable-radius offset surface is given by $[x_{min} - r, x_{max} + r] \times [y_{min} - r, y_{max} + r]$, i.e. the projected axis-aligned bounding box extended by the largest of the three radii. The geometry shader generates this rectangle consisting of two triangles and sends it to the fragment shader. In addition, the original coordinates of the triangle vertices are passed such that the link between the generated rectangle and the original triangle is known (see listing 1).

Finally, all objects on the image plane are rasterized. The resulting fragments are processed in the fragment shader. We generate a straight line whose reference points depends on the fragment position. The direction of the line corresponds to the view direction which is equal to the z-direction due to the parallel projection. Afterwards, we determine the intersection of this line with the variable-radius offset surface. If there is an intersection, the corresponding interval is saved, otherwise the fragment will be discarded (see listing 2). If we consider not only the triangular surface of a model, but the solid as such, then the front intersection of the ray with the variable-offset surface provides the offset surface, while the rear intersection defines the inset surface.

Since the triangle mesh consists of more than one triangle, we have to form the union of the resulting intersection intervals. As a data structure we propose a set of arrays with a predefined length, one for each pixel p_i, $i = 0, \ldots, m-1$. The first entry contains a mutex variable $m_i \in \{0, 1\}$ which is used for concurrency control. In the second entry we store the total amount of intervals associated with this pixel. The following elements describe depth intervals $[z_{2i}, z_{2i+1}]$, $i = 0, \ldots, n-1$, where the number of intervals n depends on the depth complexity of the object. Our algorithm works incrementally by inserting new intersection intervals into this array. Thereby, inserting a new interval $[t_1, t_2]$ means to calculate the union of this interval with the given set of ordered intervals. We perform a kind of insertion sort algorithm to find the correct positions of t_1 and t_2. Based on these positions we form the union by using shifts and deletions. This layout of the data is suitable for the GPU. In our implementation, the arrays for all pixels are combined into a single one-dimensional array, which is connected to a shader storage buffer object (SSBO). Since SSBOs are not only readable but also writable, they are well suited for our application. Unfortunately, SSBOs use incoherent memory access. As the calculation on the GPU is performed in parallel for many triangles, this must be taken into account. To avoid errors while inserting a new interval, we implemented a mutex function, which utilizes the mutexes m_i to block parts of the SSBO which are already in use. We use the GLSL function `atomicCompSwap` which exchanges a value of an array iff two given values are equal. Our mutex function uses `atomicCompSwap` to repeatedly check whether the mutex entry m_i of the current pixel array is zero. If $m_i = 0$ we set $m_i = 1$ and the insertion of a new interval can start. We take the union of the new interval with the already existing set of intervals saved in the part of the array which belongs to the actual straight line. After finishing the insertion, the intervals are disjoint again. We have to update the number of intervals and check whether it still fits the maximum number of intervals. Finally, the mutex entry must be reset, i.e. $m_i = 0$. Instead of a fixed array for each pixel, one could also use linked lists, as it is done in modern A-Buffer implementations (see e.g. [22]).

In the next section we discuss the details of determining the intersection of the straight line with the variable-radius offset surface.

3.1 Variable-radius offset surface

A straight line with reference point $p \in \mathbb{R}^3$ describing the position of a fragment and direction $u \in \mathbb{R}^3$ corresponding to the view direction is defined by

$$s(t) := p + t \cdot u,\ t \in \mathbb{R}.$$

The variable-radius offset surface of a triangle T_{abc} can be easily visualized, as shown in Figure 1 on the top and in the center. The triangle can be understood as a skeleton of the surface (see [2]). Tkach et al. [19] use this representation for hand modeling and tracking. In contrast to our work, they use an iso-level-function to describe the variable-radius offset surface which leads to complex computations. We use ray-tracing to approximate the surface instead. The final goal is to calculate the intersection of the ray $s(t)$ with the variable-radius offset surface. Therefore, we can decompose the variable-radius offset surface into simple geometric figures like spheres located at the corner points of the triangle, truncated tangential cones at the edges and triangles shifted to the tangent planes of the spheres. These simple geometric figures are displayed in the center of Figure 1 in different colors. Intersecting a ray with these geometric figures is much easier than finding an explicit form of the variable-radius offset surface. Later, the union of these different intersections is formed to obtain the correct intersection interval.

To calculate the intersection of the ray with the upper and lower triangles, we embed the triangles into the tangential planes of the spheres at the triangle vertices. The upper tangential plane can be described by

$$E_{r_+} := \{x \in \mathbb{R}^3 \mid n_+^\top (x - a) = r_a\}$$

with $n_+ \in \mathbb{R}^3$ being the normal vector of the plane. The Hesse normal form allows to determine the intersection between $s(t)$ and E_{r_+} directly by insertion. The same calculation holds to the lower triangular surface. After the intersection with the plane has been calculated, it must be checked whether it is inside the triangle. The normal vectors of the tangential planes can be determined based on the following system of equations

$$\begin{aligned} n^\top (b-a) &= r_b - r_a \\ n^\top (c-a) &= r_c - r_a \\ n^\top n &= 1 \end{aligned} \qquad (1)$$

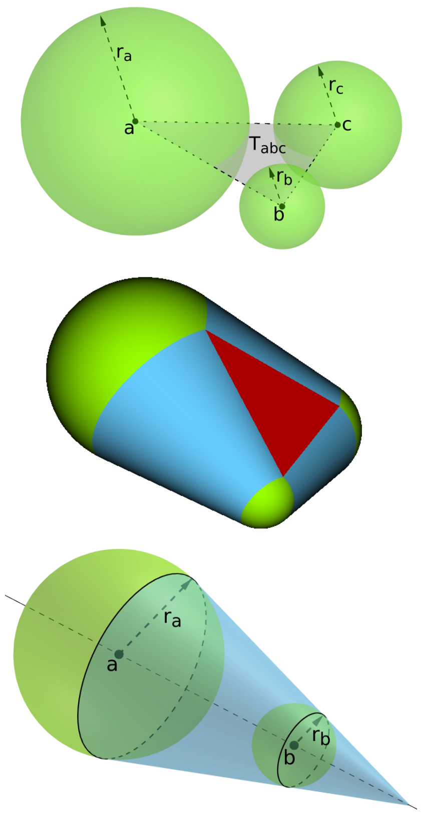


Figure 1: Top: Triangle with variable radii at the vertices. Center: Variable-radius offset surface of a triangle decomposed into spheres (green), tangential cones (blue) and tangent planes (red). Bottom: Tangential cone of the spheres at vertices a and b

by assuming that the distance of each corner point a, b, c to the tangential plane has to equal the radii r_a, r_b, r_c. System (1) leads to a quadratic equation with the two solutions n_-, n_+ for the lower and upper tangential plane.

In order to determine the intersection points of the ray $s(t)$ with the spheres placed at the corners of T_{abc}, we have to find the points on the ray whose distance from the center corresponds to the radius of the spheres.

The intersections of the ray $s(t)$ with the truncated cones at the edges of the triangle can be calculated by using the implicit form of the conical surfaces, which are tangential to the two spheres placed at the endpoints of the edge (see bottom of Figure 1):

$$(x \times (b-a) - a \times b)^2 = (r_a(x-b) - r_b(x-a))^2, \quad (2)$$

with $x \in \mathbb{R}^3$ being the points of the conical surface. The intersections of the ray $s(t)$ with the tangential cone can then be calculated by inserting $s(t)$ into (2). A solution of this quadratic equation is valid, i.e. it lies within the truncated cone, if

$$r_a(r_a - r_b) + d^\top a < d^\top x < r_b(r_a - r_b) + d^\top b.$$

with $d = b - a$.

In summary, using our method for calculating the variable-radius offset surface, only linear and quadratic equations have to be solved, which is easy to implement in a shader program. For each geometric figure we obtain an intersection interval of which we form the union.

3.2 The Minkowski Sum of a 3D mesh and a cuboid

As already mentioned, our method can be generalized to also determine the classical Minkowski sum. In the following we demonstrate this by calculating the Minkowski sum of a 3D mesh and an axis-oriented cuboid.

The variable-radius offset surface of a triangle can be easily visualized, such that a direct calculation of the intersection of a ray with the variable-radius offset surface is possible. In contrast, the decomposition of the Minkowski sum of a triangle and a cuboid into simple geometric objects is not as easy and clear. That is why we choose a different approach in this case. We use the fact that the Minkowski sum describes the subspace in which two objects collide. We assume that the triangle is fixed in space, while the cuboid can be moved along the ray. Then we determine the interval of the ray in which the cuboid center can be placed such that cuboid and triangle collide. This interval corresponds to the intersection of the ray with the Minkowski sum. Therefore we need a dynamic intersection test. In consequence of choosing the direction of the ray as the negative z-direction, the collision always occurs on the front side of the cuboid. So it is sufficient to use a rectangle instead of a cuboid, which simplifies the needed intersection test.

We use the separating axis theorem (SAT, [9]) for collision detection. It says that two convex polyhedra do not collide if it is possible to find a plane that separates them. Or in other words: Two convex polyhedra do not collide if there is an axis for which the projection of the two polyhedra do not overlap. The separating axis is perpendicular to the separating plane. It is sufficient to check only a small set of directions D which consists of the two normal vectors of the triangle and rectangle and the six pairwise cross products of the edges. Furthermore, it is also necessary to check the five normals of the edges which lie in the same plane as the triangle respectively the rectangle. This is due to the fact that when triangle and rectangle are located within the same plane, the cross product of the edges results in the zero vector.

First we assume that both the triangle and the rectangle are fixed in space and set a condition to check whether they collide. Let $o \in \mathbb{R}^3$ be the barycenter of the rectangle and $h_x, h_y \in \mathbb{R}^3$ its half-axes. We get a collision iff for all directions $d \in D$ it holds

$$d_{\min} - r \leq d^\top o \leq d_{\max} + r \tag{3}$$

with

$$\begin{aligned} r &= |d^\top h_x| + |d^\top h_y|, \\ d_{\min} &= \min\{d^\top a,\, d^\top b,\, d^\top c\} \text{ and} \\ d_{\max} &= \max\{d^\top a,\, d^\top b,\, d^\top c\}. \end{aligned}$$

In the last paragraph we assumed that both the triangle and the rectangle are fixed in space. Now, the triangle stays fixed while the rectangle can be moved along the ray $s(t) = p + t \cdot u$. Consequently, the center of the rectangle changes to $o = p + t \cdot u$, which leads to an adjusted inequality (3), i.e. for all $d \in D$ it must hold

$$d_{\min} - r \leq d^\top (p + t \cdot u) \leq d_{\max} + r. \tag{4}$$

An intersection interval is obtained for each direction to be checked. Concerning our former simplification to use a rectangle rather than a cuboid, we have to increase the intersection intervals by the length of the cuboid in z-direction. By forming the intersection of all these intervals, we obtain the interval in which no separating plane can be found. This interval is equivalent to the intersection of the ray with the Minkowski sum.

3.3 The Minkowski Sum of two 3D meshes

Finally, we can calculate the Minkowski sum of two arbitrary 3D meshes. Our approach divides one of the 3D meshes into single triangles and calculates the Minkowski sum of a 3D mesh and an individual triangle incrementally. We form the union of one resulting Minkowski sum with the ones previously calculated.

Since we use a GPU, the triangles of the 3D mesh are treated separately. Thus, we can adjust inequality (4) introduced in the last section to perform a dynamic intersection test between two triangles $T_1 = T_{a_1 b_1 c_1}$ and $T_2 = T_{a_2 b_2 c_2}$. This leads to

$$d_{\min}^{(1)} - d_{\max}^{(2)} \leq d^\top (p + t \cdot u) \leq d_{\min}^{(2)} - d_{\max}^{(1)} \tag{5}$$

with

$$\begin{aligned} d_{\min}^{(i)} &= \min\{d^\top a_i,\, d^\top b_i,\, d^\top c_i\} \quad \text{and} \\ d_{\max}^{(i)} &= \max\{d^\top a_i,\, d^\top b_i,\, d^\top c_i\},\ i = 1, 2. \end{aligned}$$

The directions $d \in D$ we have to check are the two normals of the triangles, the nine pairwise cross products of the edges and the six normals of the edges, which are lying in the same plane as the triangles to cover the degenerated case. After intersecting all these resulting intervals, we obtain the intersection interval of the ray with the Minkowski sum.

4 APPLICATIONS

After explaining the theoretical background, we now focus on some practical applications. Furthermore we will evaluate the runtime of our algorithm. All our tests were conducted on a system with an Intel Core i7 processor with 2.5 GHz, 16 GB RAM and an NVIDIA GeForce GTX 1070 GPU. The runtime of the algorithm is affected by the orientation of the individual surfaces with respect to the image plane. The reason for this is that a rectangle is generated in the geometry shader whose size corresponds to the sum of the bounding boxes of the two objects projected into the image plane. To eliminate this factor from the comparisons, the runtime is considered in relation to the number of fragments processed. For the runtime analysis of the different applications, a window size of 500 x 500 pixels is used as default. In general, doubling the window size in both the *x*- and *y*-directions leads to a factor of four of the processed fragments. The execution of our algorithm with doubled window sizes confirmed that the runtime is almost quadrupled.

4.1 Fixed radius inset/offset computation

The algorithm described in 3.1 calculates the variable-radius offset surface. Of course this approach can also be used to compute the special case of a fixed radius offset. For evaluating the approximation error of our method, we use a 3D mesh of a wooden puzzle. Figure 2 displays the original puzzle (top left) and the exact offset surface (top right) with an offset radius of 0.2. To compare our approximation error with the exact offset surface, we calculate the Hausdorff distance with a low resolution approximation respectively a higher resolution approximation (Figure 2 bottom). Therefore, the Hausdorff distances are linear interpolated between dark blue and red (see Figure 2 lower right), while dark blue parts correspond to a Hausdorff value of 0 and red parts to a Hausdorff value of at most one percent of the offset radius ($2e^{-3}$). By comparing the lower left and middle part of Figure 2, one can see, that the approximation error of our method can become arbitrary small by refining the approximation resolution.

In addition to the offset calculation, our algorithm is also suited to simultaneously calculate an inset surface. Figure 3 displays these calculations for the already mentioned wooden puzzle.

4.2 Forensics

The main application for our algorithm is facial reconstruction. Therefore, we calculate the variable-radius offset surface of a given model skull. Gietzen et al. [10] developed an automatic approach for facial reconstruction based on dense statistics of FSTT. Computer tomographic data is used to generate a 3D model of a skull. Afterwards, landmark points are mapped to a given model skull for which the average distance between the bones and the skin is known for each vertex. With respect to these vertex-radius pairs, [10] generate a sphere mesh (see [18]) which was further improved by Achenbach et al. [1].

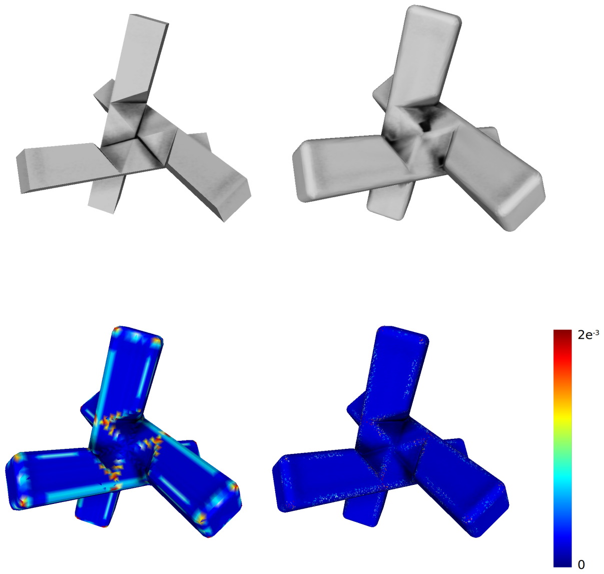


Figure 2: Original puzzle (top left), exact offset surface (top right), low poly approximation (lower left), high poly approximation (lower middle) with color bar of error measurement (lower right).

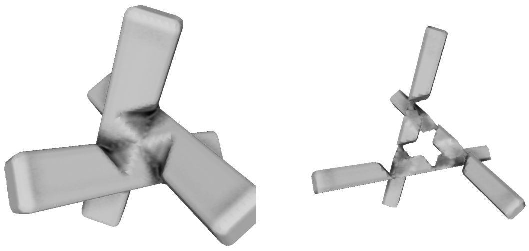


Figure 3: Offset (left) and inset (right) approximation

If FSTT data is understood as radii at the vertex points, then our approach can be used to calculate a facial reconstruction in an efficient manner by using a 3D mesh representation of the skull. Figure 4 shows the result of our algorithm. On the top the model skull is shown, while on the bottom the result of the variable-radius offset surface reconstruction using the FSTT is displayed.

To compare our approach to [1], we need to generate a new 3D mesh of the facial reconstruction. Therefore, we use a modified version of the marching cubes algorithm (see [15]). Marching cubes extracts a triangular mesh of an isosurface from a three-dimensional discrete scalar field by calculating the intersection points of this isosurface with the edges of a voxel. We adapted this

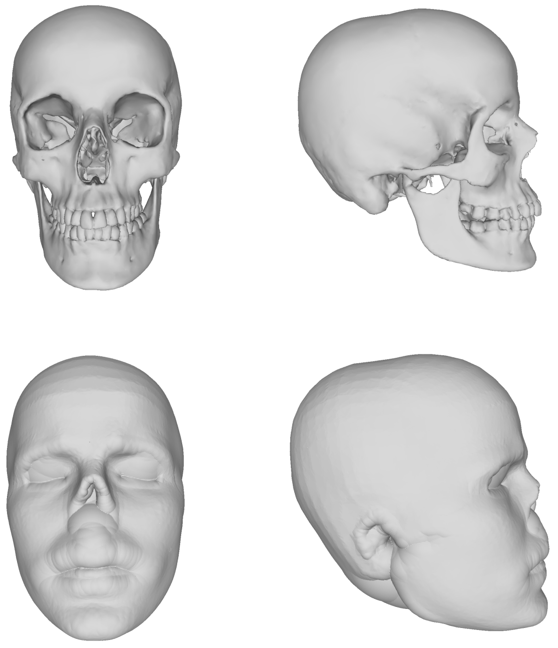


Figure 4: Facial reconstruction (bottom) of a skull (top).

algorithm so that it works with our representation. The intersection points can be read directly from our representation of the variable-radius offset surface.

The three calculations of the variable-radius offset surface need 6 seconds, while marching cubes takes an additional 18 seconds. So we need 24 seconds to perform a facial reconstruction with our approach, while the approach presented in [1] takes 67 seconds on an Intel Xeon CPU ($4 \times 3.6 Ghz$).

The runtime of the facial reconstruction depends on the chosen resolution. We choose a resolution of 1 mm per voxel, while [1] use a resolution of 2 mm per voxel. So our approach is almost 3 times faster, while at the same time allowing twice the resolution.

The upper graph in Figure 5 depicts the runtime of the fixed-radius offset surface calculation of a skull with approximately 138,000 triangular surfaces depending on the relative offset radius compared to the diagonal L_d of the bounding box. Taking a closer look at the curve, one can assume that it matches with a power function of the kind $y_i = c \cdot x_i^p$. More detailed investigations show that the exponent corresponds nearly to the value 1.5. This means that a doubling of the radius does not lead to a complete quadrature of the runtime. The lower graph in Figure 5 shows the runtime of the calculation of a variable-radius offset surface of a skull. The runtime is shown in dependency of the number of processed fragments. The curve is almost linear from about 200 fragments with a slope of 140 ms per 100 million fragments.

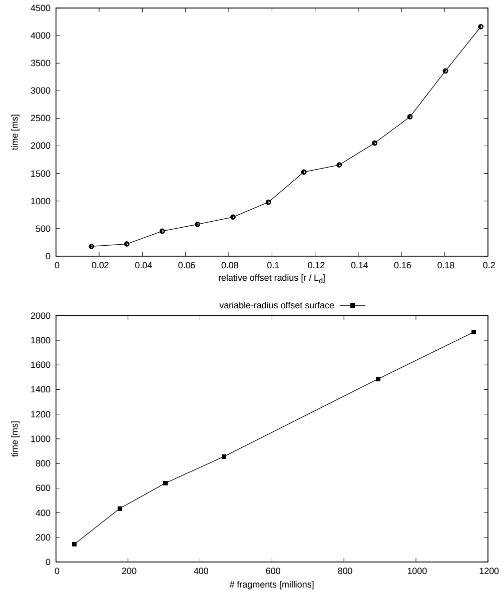


Figure 5: Top: Runtime of the fixed-radius offset surface depending on its relative radius. Bottom: Runtime of the variable-radius offset surface

4.3 Packing problems

Packing problems are a kind of optimization in which a number of objects are packed collision-free into a container. As an real world example we choose the packing of standardized cuboids into a trunk to determine its volume. There are two different standards to calculate this volume - the European and the American standard. The American standard is described in the SAE J1100 standard [17]. It requires to place a set of different sized cuboids in the trunk to determine its volume. Ding and Cagan [8] used extended pattern search to solve this problem with respect to the SAE standards.

In our work, we choose the European standard, which is described in the DIN standard 70020-1 [4] and ISO 3832 [5], but our approach can be used with the American standard as well. To determine the volume of a trunk we have to calculate the maximum amount of standardized cuboids measuring 50 x 100 x 200 mm which can be placed collision-free inside the trunk. This can be done by physically packing these cuboids or by an algorithmic approach. Eisenbrand et al. [6] developed an algorithm to solve this problem by discretizing the trunk geometry using a three-dimensional cubic grid. They used different greedy algorithms and heuristics to determine a maximum amount of axis-oriented cuboids which can be placed collision-free inside the cubic grid. In [7], they extended this approach by allowing continuous positions and orientations while using a simulated annealing algorithm to solve this problem. This randomized algorithm selects a lot of random positions inside the trunk where it tries to place the cuboids. Each time, it has to decide whether this position leads to a collision with the trunk boundary which is the bottleneck of this algorithm. This is equivalent to decide whether the given position is part of the Minkowski sum of the cuboid and the trunk. We only allow axis-oriented cuboids. Therefore, we have to calculate six different Minkowski sums, one for each different orientation. The calculation of each Minkowski sum can be done in real-time.

Using our data structure to decide whether a given position is part of the Minkowski sum corresponds to a simple look up in a list of sorted intervals. This can be done in a quasi constant time. Figure 6 shows the result of the packing algorithm using our representation of the Minkowski sum of a trunk and a standardized cuboid. Our approach can be used in all algorithms that pack axis-oriented DIN- or SAE-cuboids into a trunk. These can benefit from our method of collision detection between cuboid and trunk geometry.

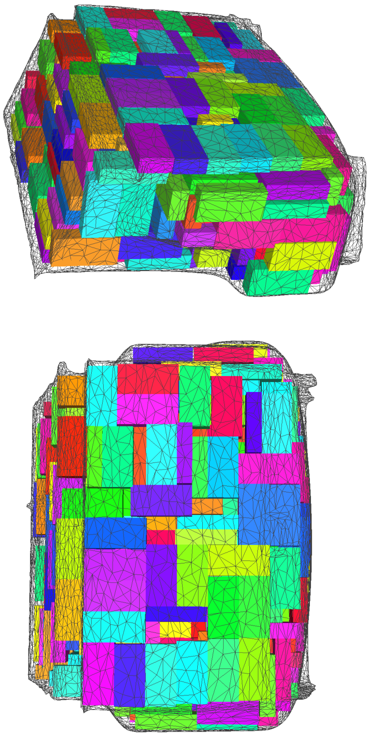


Figure 6: Result of a packing algorithm based on a trunk geometry

5 CONCLUSION AND FUTURE WORK

In this paper we presented a new method to approximate the variable-radius offset surface. Our implementation utilizes the GPU, which, due to its parallel functionality, is well-suited to perform similar calculations simultaneously. We use a set of straight lines over a two-dimensional grid to represent the variable-radius offset surface. Applying our algorithm to determine three orthogonal representations, we are able to generate a new triangular mesh of the variable-radius offset surface by adapting the marching cubes algorithm. Because the focus of our work was on developing an algorithm for calculating the variable-radius offset surface, the marching cubes algorithm is not optimized yet. Optimizing the marching cubes algorithm contains a large potential of speed improvements and could be done in further research. Our algorithm also allows the fast calculation of classical Minkowski sums between a triangular mesh and another object with a relatively small description complexity. Further improvements could focus on optimizing the implementation for general triangle meshes.

We demonstrated the value of our algorithm on three different applications. First, we applied our algorithm to offset and inset computation and showed, that the approximation error of our method can become arbitrarily small. Second, we demonstrated how to use our method for facial reconstructions in a forensic context. We compared our method with [1] and demonstrated that it provides a clear speed advantage while allowing a higher resolution. Third, we apply our algorithm to solve packing problems. As an example we determine the trunk volume by packing axis-oriented standardized cuboids. Therefore, we use the the Minkowski sum representation presented here for collision detection in quasi constant time. With the approach presented here we can calculate the Minkowski sum of the trunk geometry and axis-oriented cuboids in a few milliseconds and use them to pack standardized cuboids into a trunk without collision with the boundary.

6 REFERENCES

[1] Jascha Achenbach, Robert Brylka, Thomas Gietzen, Katja zum Hebel, Elmar Schömer, Ralf Schulze, Mario Botsch, and Ulrich Schwanecke. A Multilinear Model for Bidirectional Craniofacial Reconstruction. In Puig Puig et. al., editor, *Eurographics Workshop on Visual Computing for Biology and Medicine*, pages 67–76. The Eurographics Association, 2018.

[2] Jules Bloomenthal and Ken Shoemake. Convolution surfaces. *SIGGRAPH Comput. Graph.*, 25(4):251–256, July 1991.

[3] Marcel Campen and Leif Kobbelt. Polygonal boundary evaluation of minkowski sums and swept volumes. *Computer Graphics Forum*, 29(5):1613–1622, 2010.

[4] Deutsches Institut fuer Normung e.V. DIN 70020 part 1, road vehicles; automotive engineering; dimensional terms, 02 1993.

[5] Deutsches Institut fuer Normung e.V. ISO 3832, passenger cars - luggage compartments - method of measuring reference volume, 06 2002.

[6] Quan Ding and Jonathan Cagan. Automated trunk packing with extended pattern search. *SAE Technical Papers*, 03 2003.

[7] Friedrich Eisenbrand, Stefan Funke, Andreas Karrenbauer, Joachim Reichel, and Elmar Schömer. Packing a trunk - now with a twist! *International Journal of Computational Geometry & Applications*, 17(05):505–527, 2007.

[8] Friedrich Eisenbrand, Stefan Funke, Joachim Reichel, and Elmar Schömer. Packing a trunk. In *Algorithms - ESA 2003, 11th Annual European Symposium, Budapest, Hungary, September 16-19, 2003, Proceedings*, pages 618–629, 2003.

[9] Christer Ericson. *Real-time collision detection.* Morgan Kaufmann, 2007.

[10] Thomas Gietzen, Robert Brylka, Jascha Achenbach, Katja zum Hebel, Elmar Schömer, Mario Botsch, Ulrich Schwanecke, and Ralf Schulze. A method for automatic forensic facial reconstruction based on dense statistics of soft tissue thickness. *PLOS ONE*, 14(1):1–19, 01 2019.

[11] Peter Hachenberger. Exact minkowksi sums of polyhedra and exact and efficient decomposition of polyhedra into convex pieces. *Algorithmica*, 55(2):329–345, 2009.

[12] Min-Ho Kyung, Elisha Sacks, and Victor Milenkovic. Robust polyhedral minkowski sums with gpu implementation. *Computer-Aided Design*, 67:48–57, 2015.

[13] Wei Li and Sara McMains. Voxelized minkowski sum computation on the gpu with robust culling. *Computer-Aided Design*, 43(10):1270–1283, 2011.

[14] Jyh-Ming Lien. A simple method for computing minkowski sum boundary in 3d using collision detection. In *Algorithmic foundation of robotics VIII*, pages 401–415. Springer, 2009.

[15] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH Comput. Graph.*, 21(4):163–169, August 1987.

[16] Lin Qun and J. G. Rokne. Variable-radius offset curves and surfaces. *Mathematical and Computer Modelling*, 26(7):97 – 108, 1997.

[17] Society of Automotive Engineers. SAE J1100, motor vehicle dimensions, 11 2009.

[18] Jean-Marc Thiery, Émilie Guy, and Tamy Boubekeur. Sphere-meshes: Shape approximation using spherical quadric error metrics. *ACM Transactions on Graphics*, 32(6):1 – 12, 2013.

[19] Anastasia Tkach, Mark Pauly, and Andrea Tagliasacchi. Sphere-meshes for real-time hand modeling and tracking. *ACM Trans. Graph.*, 35(6), November 2016.

[20] Gokul Varadhan and Dinesh Manocha. Accurate minkowski sum approximation of polyhedral models. In *12th Pacific Conference on Computer Graphics and Applications, 2004. PG 2004. Proceedings.*, pages 392–401. IEEE, 2004.

[21] Charlie CL Wang and Dinesh Manocha. Gpu-based offset surface computation using point samples. *Computer-Aided Design*, 45(2):321–330, 2013.

[22] Jason C Yang, Justin Hensley, Holger Grün, and Nicolas Thibieroz. Real-time concurrent linked list construction on the gpu. In *Computer Graphics Forum*, volume 29, pages 1297–1304. Wiley Online Library, 2010.

A SHADER IMPLEMENTATION

```
#version 450 core
layout (triangles) in;
layout (triangle_strip, max_vertices = 6) out;
layout (std430, binding=3) buffer data_buffer {
    float radians[]; };

in VS_OUT {
    int index;
} vdata[];

out GS_OUT {
    vec3 vertices[3];
    float id[3];
} data;

// uniform variables: zoom, uMMat, uVMat, uPMat

void main() {
    vec4 hp[3];
    vec3 p[3];
    for(int i=0; i<3; i++) {
        data.vertices[i] = (uMMat * gl_in[i].gl_Position).xyz;
        data.id[i] = vdata[i].index;
        hp[i] = uPMat * uVMat * uMMat * gl_in[i].gl_Position;
        p[i] = hp[i].xyz/hp[i].w; }

    float radius_ = max(radians[vdata[0].index], max(radians[vdata[1].index], radians[vdata[2].index]));

    vec4 radius = uPMat * vec4(zoom*radius_, zoom*radius_, 0, 1);

    float min_x = min(p[0].x, min(p[1].x, p[2].x));
    float max_x = max(p[0].x, max(p[1].x, p[2].x));
    float min_y = min(p[0].y, min(p[1].y, p[2].y));
    float max_y = max(p[0].y, max(p[1].y, p[2].y));

    vec4 s = vec4(min_x - radius.x, min_y - radius.y, 0, 1);
    vec4 t = vec4(max_x + radius.x, min_y - radius.y, 0, 1);
    vec4 u = vec4(min_x - radius.x, max_y + radius.y, 0, 1);
    vec4 v = vec4(max_x + radius.x, max_y + radius.y, 0, 1);

    gl_Position = s; EmitVertex();
    gl_Position = t; EmitVertex();
    gl_Position = u; EmitVertex();
    EndPrimitive();

    gl_Position = t; EmitVertex();
    gl_Position = v; EmitVertex();
    gl_Position = u; EmitVertex();
    EndPrimitive();
}
```

Listing 1: Essential parts of the geometry shader

```
#version 450 core
layout (std430, binding=1) buffer offset_data {
    int intervals[]; };
layout (std430, binding=3) buffer data_buffer {
    float radians[]; };
layout (std430, binding=5) buffer shader_error {
    int error[4]; };

in GS_OUT {
    vec3 vertices[3];
    float id[3];
} data;

// uniform variables: uMMat, uVMat, uPMat, uInvVMat, uInvPMat, width, height, lod, decimalPrecision
vec3 intercept_points[7], normals[7]; // ray surface intersections

void mutex(int t1, int t2, uint i) {
    int max_value = 5000;
    bool done = false;

    while (!done && max_value-- > 0) {
        if (atomicCompSwap(intervals[i], 0, 1) == 0) {
            // insert interval [t1, t2] into interval list at index i, see section 3
            save_interval(t1, t2, i);
            atomicExchange(intervals[i], 0);
            done = true; }
    }
    if (max_value == 0)
        atomicExchange(error[0], 1);
}

int check_t(vec3 t[7], uint i) {
    bool intersection = false;
    int idx = -1;

    for (int i = 0; i < t.length(); i++) {
        if (t[i].z != -1) {
            intersection = true;
            break; }
    }
    if (intersection == false)
        discard;

    float t_small = infinity;
    float t_big = -infinity;

    for (int j = 0; j < t.length(); j++) {
        if (t[j].z == -1)
            continue;
        t_small = min(t_small, t[j].x);
        if (t[j].y > t_big) {
            t_big = t[j].y;
            idx = j; }
    }

    if (t_small > t_big)
        discard;

    mutex(int(floor(t_small * decimalPrecision)), int(ceil(t_big * decimalPrecision)), i);

    return idx;
}

void main() {
    // Build triangle normal ------------------------
    vec3 e0 = data.vertices[1] - data.vertices[0];
    vec3 e1 = data.vertices[2] - data.vertices[0];
    vec3 e2 = data.vertices[1] - data.vertices[2];
    vec3 n_triangle = normalize(cross(e0, e1));

    vec4 u1 = uInvVMat * uInvPMat * vec4(0, 0, -1, 0);
    vec3 u = normalize(u1.xyz); // view direction

    uint posX = uint(gl_FragCoord.x);
    uint posY = uint(gl_FragCoord.y);
    uint idx = (posY * width + posX) * lod * 2;

    vec4 p1 = uInvVMat * uInvPMat * vec4((2.0f * posX)/width - 1, (2.0f * posY)/height - 1, 0, 1);
    vec3 p = p1.xyz / p1.w;

    // Intersections ---------------------------
    vec3 tmp[7]; // store depth values of intersections
    for (int i = 0; i < tmp.length(); i++) {
        tmp[i] = vec3(0, 0, -1);
        intercept_points[i] = vec3(0, 0, 0);
    }

    // see section 3.1 for detailed intersection tests
    intersect_triangle(p, u, n_triangle, tmp[0]);

    intersect_sphere(p, u, 0, tmp[1]);
    intersect_sphere(p, u, 1, tmp[2]);
    intersect_sphere(p, u, 2, tmp[3]);

    intersect_cone(p, u, 0, 1, tmp[4]);
    intersect_cone(p, u, 2, 0, tmp[5]);
    intersect_cone(p, u, 1, 2, tmp[6]);

    // union of intervals; find index of nearest point in view direction
    int min_idx = check_t(tmp, idx);

    vec4 ndc = uPMat * uVMat * vec4(intercept_points[min_idx], 1);
    gl_FragDepth = (1 + ndc.z)/2;

    vec3 normal_ = normalize(normals[min_idx]);
    vec3 lDir = normalize(u); // light direction
    FragColor = vec4(max(dot(lDir, normal_),0.0)*vec3(0.94), 1.0);
}
```

Listing 2: Essential parts of the fragment shader

Learning Capacity in Simulated Virtual Neurological Procedures

Mattia Samuel Mancosu
Department of Computer Science, Liverpool John Moores University
Brownlow Hill
L3 5UG, Liverpool,
United Kingdom
m.s.mancosu@2019.ljmu.ac.uk

Silvester Czanner
Department of Computer Science, Liverpool John Moores University
Brownlow Hill
L3 5UG, Liverpool,
United Kingdom
s.czanner@ljmu.ac.uk

Martin Punter
Senior Medical Officer
Capital and Coast District Health Board
69 Riddiford Street
Wellington, New Zealand
martin.punter@ccdhb.org.nz

ABSTRACT

In recent years, the use of Virtual Reality (VR) has skyrocketed in many fields. From videogames to education, the popularity of this technology is well known also thanks to its affordability and the growing number of available content. An immersive approach and the correct use of gamification have been proven to be valuable tools for learning in different fields. There are many examples of the adoption of VR application also in the healthcare setting not only for learning purposes but also for patients' rehabilitation. This paper is intended to list some existing works and then present our project for an immersive serious game with the aim of enhancing the training of healthcare workers for the assessment of stroke patients.

Keywords
Virtual Reality, Serious Games, Gamification, Healthcare, Stroke, Evaluation, Assessment, Training

1 INTRODUCTION

Real-life simulation is frequently used in clinical practice for training in the United Kingdom and further afield. One of the most successful programmes is with simulation training for Advanced Life Support where regular highly structured practice improves team performance. Simulation training is also taking on an increasing role in the training of medical students, junior doctors, and Allied Health Professionals in day to day work, whether for specific procedures or specific team practices. At Salford Royal NHS Foundation Trust (SRFT) has been recently introduced real-time simulation training for the stroke team, using a mannequin and running through clinical scenarios relating to acute stroke treatment.

It can be increasingly a challenge, however, in the National Health Service, to find appropriately convenient times to bring whole teams together in one place for such training and the opportunity for VR training may improve staff confidence in advance of real-time simulation training, or clinical practice, and would allow training to be delivered simultaneously across multiple sites. Whilst serious games or VR have been described as used in the rehabilitation of stroke patients (Cochrane review 2015, Laver et al) to our knowledge no publications are describing the use of VR as a tool for systematic training of healthcare professionals involved in the care of patients with acute stroke.

2 VIRTUAL REALITY

For Soysa et al. Virtual Reality is intended as a Human-Computer interface where a computer system generates a three dimensional, sensory, immersing environment that responds in an interactive way to the behaviour of the user [1]. The purpose of their research was to create a framework for VR and Virtual Instrumentation for educational purposes, performing an evaluation of an application based on the response of users to haptic feedback. Until a few years ago VR was not immediately associated with the immersive part of the experience. Only recently have head-mounted displays (HMD) become increasingly popular and led to a natural association with VR. Previously, many VR projects orbited around the concept of Collaborative Virtual Environments (CVE), defined by Normand et al. as the use of natural spatial metaphors, together with the integration of participants and data within the same and common spatial frame of reference [2]. For example, Nakai et al. realised a network infrastructure for peer to peer distributed computation on a collaborative virtual envi-

https://doi.org/10.24132/CSRN.2020.3001.13

ronment [3]. With the proper use of Level Of Details function for 3D elements and the correct distribution of the contents within the nodes of the peer to peer network, they achieve a good level of usability and scalability of the CVE system. There are many articles concerning rules and guidelines for the proper use of VR and CVE. Levels of detail is considered a crucial feature when designing a VR application also by Seo and Kim, as they help cope with problems related to bigger environments or incrementally complex scenes [4]. In 1999 Noll et al. described the interaction between users inside a shared virtual environment and how this interaction could enhance their work. This research covered also the areas of distributed simulation, agents in virtual environments, and networking for large-scale virtual environments [5]. Kuutti et al. stated that virtual prototypes can be used for usability testing and the quality of environment and models are crucial for the success of the testing [6]. In fact, Yen and NG also showed that Virtual Objects are successful tools for Virtual Environments if they are properly implemented and can be far more useful than simple images [7]. Furthermore, a study conducted by Drettakis et al. illustrated an evaluation approach for real-world applications in the context of an urban planning project, demonstrating the relevance of both audio and visual realism, such as 3D sound, shadows, sun coverage, vegetation, and crowds [8]. Many evaluation guidelines on VR application for multi-disciplinary activities are addressed by Livatino and Koffel [9]. Their two pilot studies successfully proved the correctness and the reasonableness of those guidelines for students and non-experts. More recently, Bouville et al. presented a framework for developing an interactive and collaborative virtual environment. Their system is based on a *relation engine* module and a *collaborative interaction engine* module. The first one manages the relations between the objects of the environment and the collaborative interaction engine manages how users can collaboratively control objects [10].

2.1 Immersive Virtual Reality

The immersive approach to Virtual Reality extended the possible applications, the type of interactions and the quality of the experience. Corrêa et al. researched how a realistic 3D environment in conjunction with immersive VR is perceived. Performing an evaluation on interactive archaeometry, they concluded that it is possible to represent VE realistically as the real one [11]. The use of HMD has led to multiple new interaction possibilities, reflected in the number of new studies concerning movement, user interface and usability. For example, Gebhardt et al. performed a study focused on different entry selection methods for hierarchical pie menus. They tested the use of input devices different from the standard keyboard and mouse, and they concluded that the pick-ray-based entry selection metaphor is the most suitable for this case of study [12]. In 2014 Emma et al. described the nature of usability issues in immersive CVE. The evaluation is performed inside a framework where errors observed are interpreted as rule-based mistakes. The errors are mainly connected with tactual and/or postural issues. In fact, the lack of tactual feedback when touching an object and the sitting position of the participants had contributed to thwarting the realism of the actions [13].

3 SERIOUS GAMES

VR and teaching is a well known successful combination, there are countless educational and industrial application instances. For the field of Mechanical Engineering, Muller et al. presented a preliminary study on the use of consumer-grade VR devices, describing how to design an immersive learning game and how to evaluate the user experience. The first exploratory experiment showed a high interest from teachers and students, laying the foundations for future improvement of the training in the Mechanical Engineering field [14]. Concerning learning games and safety, in 2017 Zhang et al. designed a framework for fire safety education implemented according to the combination of relevant fire safety education theory and immersive VR technology. Using an HTC VIVE helmet equipment for the user test, they again proved the effectiveness of this technology to help students learn fire safety knowledge, master fire safety skills and improve fire safety education effect [15]. Another example of immersive experience in safety education is the game developed by Vega et al. in 2017. The design is focused on a controlled and engaging environment to practice and learn how to safely control a campfire. They received positive feedback in many competitions, although they still have to perform a proper evaluation of the game [16]. On the subject, Andrès et al.affirmed that *"to demonstrate the effectiveness of a serious game, a holistic evaluation process is required"*. Their first objective was to evaluate a serious game called Timemesh and to show the importance of adequate evaluation process [17]. An in-depth exemplar designed to illustrate how VR can be used to promote aspects of Deeper Learning[18] for high school students is provided in a paper by Southgate. The study shows how VR can be used for self-directed, collaborative and imaginative learning [19]. Zhang et al. explored the possible learning outcomes derived by different the level of interaction in a single VR learning application. Although they were not able to find a significant correlation between level of interaction and learning outcome, the interaction has been proven to be a key factor for engagement and their data suggests that "interactivity-triggered concept visualisation and animation are more effective for learning than automated visualisation and animation" [20].

3.1 Serious Games in Healthcare

Games are often used for training healthcare workers and during patients rehabilitation. The proper application of gamification techniques can be crucial to both categories. In this regard, Ribeiro et al. in 2019 presented a system to practice dental anaesthesia techniques. Their serious games allow the user to practice with or without tactile or visual aids thanks to a haptic interface [21]. Also, Ushaw et al. proved the help given by serious games for health. Their game for measuring the stereoacuity of young children has been tested by an initial user group and was proven successful, although a testing protocol and rapid analysis of the data is needed [22]. In this paper, our main research objective is immersive VR and serious games in healthcare, more specifically in the rapid assessment and treatment of acute stroke, are relatively common emergency neurological disease. In 2006 Kaul combined gaming technology with brain wave research to create a gaming environment that functions as a neurological learning tool, explaining that mental states can be focused and the adoption of a content-driven environment can help the ability to perceive information [23]. Alankus et al. cited the importance of multiple repetitions of motion with the affected limbs in patients paralysed by strokes. Considering the high rate of unmotivated patients, home-based stroke rehabilitation games may help motivate stroke patients to perform the necessary exercises to recover. Starting from the previous considerations, Alankus et al. developed several games for coping with different necessities for rehabilitation such impossibility of the use of hands and the support for multiple methods of user input [24]. Those games are not realised using VR techniques, however, they proved their value from a therapeutic perspective even with a limited number of tested patients. Choi, on the other hand, presents the application of VR technology in wound care training. The proposed application simulates the steps for changing a wound dressing with the help of visual cues and texts. Although this study is not concluded yet and comprehensive evaluation is still needed, the system has the promising benefit of facilitating self-learning at own pace while saving the cost of wound dressing materials [25].

4 GAMIFICATION

Serious Games share with Gamification a similar intent, which is learning having fun to improve the effectiveness of teaching. Gamification, however, is the application of game elements (such as badges and leader boards) to other areas of activity to increase effectiveness and engagement. We are then talking about general-purpose applications instead of games, which have fun and entertainment as primary objectives. Marczewski defines Gamification as "the use of game design metaphors to create more game-like and engaging experiences."[26] For instance, Pastushenko et al. developed a tool for the conceptual learning of electric and electronic circuits. In this case, the student has to modify some values on the proposed circuits in order to obtain the correct voltage across the given components and let the game character "surf" to the next circuit [27]. The research proposed by Coull et al. makes use of Gamification for developing an application capable of simulate cybercrimes and provide training to police officers. A virtual environment is adopted to host a scenario where the officers can explore a specific crime scene and, by inspecting elements such as smartphones and laptops, give feedback with the appropriate course of action. The results have demonstrated the potential of this type of training considering also a significantly reduced cost with respect to expensive classroom-based sessions [28].

4.1 Gamification in Healthcare

Gamification is also a tool used for healthcare applications. Jacobs et al. designed a game for stroke survivors aimed to make the rehabilitation process more enjoyable. Their application involves the use of common objects and a multi-touch screen to perform arm-hand exercises in order to restore physical and cognitive capabilities. The results of the test conducted on two patients showed promising therapy results with a noticeable improvement of their arm-hand functionality [29]. An online game for medical education on insulin therapy has been objective of study for Diehl et al. The game is a tool for the education of doctors in the treatment of patients with diabetes mellitus and Diehl et al. collected results from 134 of them after 21 days of interaction with the game. Two groups were formed to study the difference between who played the game (game group) and the control group. The absolute increase in competence score and the frequency of subjects who achieved a high score in the post-intervention assessment was considerably higher in the game group than in the control group [30]. Vascular Invaders is a web-based study aid with game features adopted by Gauthier et al. to perform a comparative study between regular learning and eLearning on a group of medical anatomy students. Their results showed the necessity for a more exploratory study of the subject considering the promising results [31]. Boada et al. analysed the impact of using LISSA, a game to complement CPR teaching, after theory introduction and before laboratory practice on 109 nursing undergraduate students. Even then, the use of Gamification helped to engage the students and better results were obtained with respect to students that only read theory material [32].

5 SIMULATED NEUROLOGICAL TRAINING

Neurological procedures are very complex. It is almost impossible to simulate the whole procedure for several reasons. Even there is a guidance and exacts assessment instructions for the neurologists, each procedure is different in terms of patients medical conditions. Based on neurological advice we have identified a clearly defined project, of manageable size and complexity that established and rigorously test the necessary technical capabilities in an experimental setting.Real-life simulated training is well established in certain medical specialties (typically emergency department, CPR training, ICU medicine) and is being increasingly used in other specialites. There is a a great deal of interest in the use of simulation for the assessment and treatment of acute stroke as it is a highly protocolised and time sensitive treatment and with widely available stroke specialist teams, developing efficient team working improves outcomes for patients. The overall goal in this project was to design and develop a virtual stroke simulator with software support to use for simulation of acute stroke assessment and treatment. The aim is educating healthcare professional involved in the management of acute stroke and provide insight for young and elderly people on the effects and symptoms of a stroke and what to expect in the hospital setting. The proposed virtual stroke simulator is built on a spectrum of information acquired by a detailed literature search, input from clinical studies, and through discussions with clinical neurologists. We are using a standardised approach to build the simulator based on international treatment guidelines and structured rating scales of stroke severity and inclusion/exclusion criteria for the various treatments of stroke.

One of the main issues with the training for healthcare professionals is availability and time restriction. It can be a challenge, particularly in the modern healthcare setting, to gather a learning group in one place for training. Therefore being able to access, either individually or as a team, an online simulation training package could offer a solution. In the setting of busy stroke units, the opportunity to train in a virtual environment with one or more team members participating in a VR game would provide enhances opportunities for practice and refining teamwork. Furthermore, the standard training procedure involves the use of mannequins which can be suitable for helping with only a few of the many steps needed for assessing a real stroke case. For example, facial palsy, aphasia and limb ataxia cannot be shown using a mannequin and usually, they are simulated by a volunteer. By developing such simulator we aim to improve the effectiveness of the training.

5.1 Acute Stroke Assessment

A stroke is a medical condition where the blood supply of a part of the brain is cut off and it is usually diagnosed by doing physical tests and studying images of the brain produced during a scan [33]. Due to the complexity of the assessment, our simulation is limited to the assessment of the severity grade of the stroke and it will be based on physical movement, facial expressions, speech and other actions necessary for the evaluation but not currently possible to replicate using a mannequin. We decided to simulate the National Institutes of Health Stroke Scale (NIHSS) procedure. "The NIHSS is a 15-item neurologic examination stroke scale used to evaluate the effect of acute cerebral infarction on the levels of consciousness, language, neglect, visual-field loss, extraocular movement, motor strength, ataxia, dysarthria, and sensory loss. A trained observer rates the patent's ability to answer questions and perform activities. Ratings for each item are scored with 3 to 5 grades with 0 as normal, and there is an allowance for untestable items. The single patient assessment requires less than 10 minutes to complete. The evaluation of stroke severity depends upon the ability of the observer to accurately and consistently assess the patient."[34]

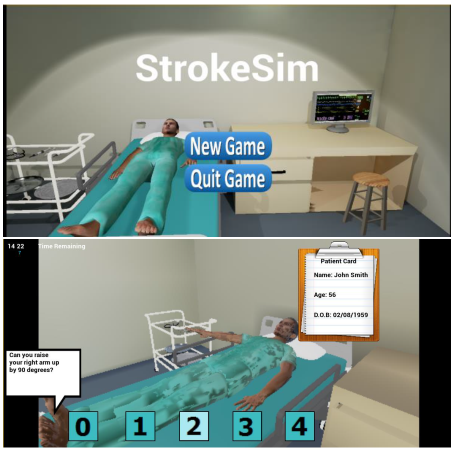
5.2 Pilot Studies

The predecessors of our project were two pilots based on the students' research projects. The objective was to try to simulate a stroke case assessment by showing a few animations related to simple evaluable actions. For example arm or leg lifting [Figure 1]. We have evaluated the simulation with the medical community and a group of 5 neurologists from Salford Royal NHS Foundation Trust. Although the pilot applications were implemented successfully, and the novel approach well received by the physicians who participated, the simulations were far from being suitable for the trial because of the absence of a proper procedure in developing, testing and case studies.

During this preliminary testing, feedback regarding the application included a need for better animations. It was clear for each respondent that the game had a good base from which to build upon but it needed more work and animations need to be improved in their performance for medical experts to recognise their use. The variety of the animations was also a problem. As we stated in the previous paragraph, in order to correctly assess a stroke we need to analyse 15 different steps of the assessment and this pilot projects covered only simple actions like arm and leg lift.

5.3 Future Work

We are currently working on the implementation of all the 15 steps of the assessment. Each step is designed to



Patient Card
Name: John Smith
Age: 56
D.O.B: 02/08/1959

Can you raise your right arm up by 90 degrees?

Figure 1: Screenshots from one of the pilot projects. In the second image, the patient is asked to raise the arm to 90 degrees since the movement is not fully achieved the assigned rank for that task is 2.

be as realistic as possible. The level of realism is important especially for the animations of movement and facial expressions. These features in conjunction with the use of Virtual Reality, are the key milestones of our simulation, which is implemented in the form of a serious game [Figure 2]. Once the game has been released, the next step is to perform various tests aimed at evaluating different aspects of the simulation. We will carry out an evaluation for the VR learning environment to assess the realism and the accuracy of the game, evaluation of the effectiveness end the usability of our tool and evaluation based on the comparison between the results of the standard training and our training method. This last step will be focused on analysing common factors between the two methods like the time spent on each step of the assessment throughout the training, the number of correct results or the number of training sessions needed to learn how to correctly assess the stroke. All those metrics will be considered as important markers needed to affirm or dispute the efficiency of our project as a training tool.

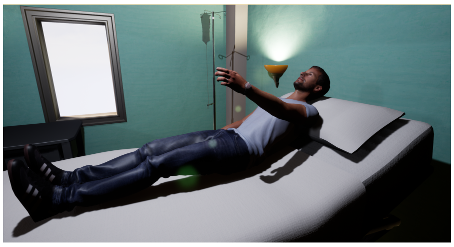


Figure 2: Screenshot from the new concept. The patient is asked to raise the arm to 45 degrees and if he will be capable of keeping the arm straight with no drift for 10 seconds the score for this step of the assessment will be 0 out of 4, where a higher score means a more severe condition of the patient.

6 CONCLUSION

The first part of this paper aimed to conduct a detailed literature survey on the virtual environment and serious games used in healthcare. We listed some existing methods for simulations and serious games used in neurological clinical practice. We explained the concept of Virtual Reality and the differences with Immersive Virtual reality. We also described several publications in the field of Serious Games and finally Serious Games in healthcare. We can affirm that Virtual Reality and Serious Games represent an efficient combination when dealing with educational problems. Their value for learning and rehabilitation purposes is vastly acknowledged and there is still the potential for improvement. Many games are used for stroke rehabilitation but we have not found any example of VR Serious Games for the training of doctors and neurologists in the same clinical field. Based on this necessity and starting from a favourable state of the art on the subject we showed two successful pilot studies and we proposed a possible new solution to the absence of a reliable training tool for the assessment of acute stroke patients.

7 ACKNOWLEDGEMENT

We would like to thank Jamie Tomlin and Adam Maxwell for their contribution to the pilot projects and to Salford Royal Foundation Trust for helping with the design and evaluation of the pilot studies.

8 REFERENCES

[1] Soysa, Ramesha & Parthiban, R & How, Khoo. (2012). Interaction-Oriented System for VITAL Framework. Proceedings - 2012 IEEE 4th International Conference on Technology for Education, T4E 2012. 192-195. 10.1109/T4E.2012.44.

[2] Normand, Vronique & Babski, Christian & Benford, Steve & Bullock, Adrian & Carion, Stphane & Frécon, Emmanuel & Kuijpers, Nico & Thalmann, Nadia & Slater, Mel & Smith, Gareth & Steed, Anthony & Thalmann, Daniel & Tromp, Jolanda & Usoh, Martin & Van Liempd, Gidi & Harvey, John. (1999). The COVEN project: exploring applicative, technical and usage dimensions of collaborative virtual environments.

[3] Nakai, Y & Kamon, N & Shibata, Y. (2006). Collaborative Virtual Environment Considering Scalability and Usability. 2006. 121 - 121. 10.1109/MDM.2006.46.

[4] Seo, Jinseok & Kim, Gerard. (2002). Design for Presence: A Structured Approach to Virtual Reality System Design. Presence. 11. 378-403. 10.1162/105474602760204291.

[5] Noll, S & Paul, C & Peters, R & Schiffner, N. (1999). Autonomous agents in collaborative

virtual environments. 208 - 215. 10.1109/ENABL.1999.805203.

[6] Kuutti, Kari & Battarbee, Katja & Säde, Simo & Mattelmäki, Tuuli & Keinonen, T & Teirikko, T & Tornberg, A.-M. (2001). Virtual Prototypes in Usability Testing. Hawaii International Conference on System Sciences. 5. 5029. 10.1109/HICSS.2001.926545.

[7] Yen, Benjamin & Ng, Kenny. (2007). Virtual Objects in Electronic Catalogs: A Human-Computer Interface Issue. Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on. 37. 599 - 608. 10.1109/TSMCA.2007.897698.

[8] Drettakis, George & Roussou, Maria & Reche Martinez, Alex & Tsingos, Nicolas. (2007). Design and Evaluation of a Real-World Virtual Environment for Architecture and Urban Planning. Presence. 16. 318-332. 10.1162/pres.16.3.318.

[9] Livatino, Salvatore & Koffel, C. (2007). Handbook for Evaluation Studies in Virtual Reality. 1 - 6. 10.1109/VECIMS.2007.4373918.

[10] Rozenn Bouville, Valérie Gouranton, Thomas Boggini, Florian Nouviale, Bruno Arnaldi. #FIVE : High-Level Components for Developing Collaborative and Interactive Virtual Environments. Proceedings of Eighth Workshop on Software Engineering and Architectures for Realtime Interactive Systems (SEARIS 2015), conjunction with IEEE Virtual Reality (VR), Mar 2015, Arles, France. (hal-01147734)

[11] Corrêa, Ana Grasielle & Borba, Eduardo & Lopes, Roseli & Zuffo, Marcelo & Araujo, Astolfo & Kopper, Regis. (2017). User experience evaluation with archaeometry interactive tools in Virtual Reality environment. 217-218. 10.1109/3DUI.2017.7893349.

[12] Gebhardt, Sascha & Pick, Sebastian & Leithold, Franziska & Hentschel, Bernd & Kuhlen, Torsten. (2013). Extended Pie Menus for Immersive Virtual Environments. IEEE transactions on visualization and computer graphics. 19. 644-51. 10.1109/TVCG.2013.31.

[13] Emma, Lógó & Peter Hamornik, Balazs & Köles, Máté & Hercegfi, Karoly & Tóvölgyi, Sarolta & Komlodi, Anita. (2015). Usability related human errors in a collaborative immersive VR environment. 5th IEEE International Conference on Cognitive Infocommunications, CogInfoCom 2014 - Proceedings. 243-246. 10.1109/CogInfoCom.2014.7020454.

[14] Muller, Nicolas & Panzoli, David & Michel, Galaup & Lagarrigue, Pierre & Jessel, Jean-Pierre. (2017). Learning mechanical engineering in a virtual workshop: A preliminary study on utilisability, utility and acceptability. 55-62. 10.1109/VS-GAMES.2017.8055811.

[15] Zhang, Kun & Suo, Jintao & Chen, Jingying & Liu, Xiaodi & Gao, Lei. (2017). Design and Implementation of Fire Safety Education System on Campus based on Virtual Reality Technology. 1297-1300. 10.15439/2017F376.

[16] Vega, Jessica & Rose, Sophia & Eckhardt, Christian & Tahai, Liudmila & Humer, Irene & Pietroszek, Krzysztof. (2017). VR wildfire prevention: teaching campfire safety in a gamified immersive environment. 1-2. 10.1145/3139131.3141218.

[17] Andrés, P.M., Serón, F.J., López-Moreno, J., & Carvalho, C.V. (2014). TimeMesh: Producing and Evaluating a Serious Game. Interacción.

[18] American Institutes for Research. Does deeper learning improve student outcomes? Results from the study of deeper learning: Opportunities and outcomes. August 2016. Retrieved https://www.air.org/sites/default/files/Deeper-Learning-Summary-Updated-August-2016.pdf

[19] Southgate, Erica. (2019). Virtual Reality for Deeper Learning: An Exemplar from High School Science. 1633-1639. 10.1109/VR.2019.8797841.

[20] L. Zhang, D. A. Bowman and C. N. Jones, "Exploring Effects of Interactivity on Learning with Interactive Storytelling in Immersive Virtual Reality," 2019 11th International Conference on Virtual Worlds and Games for Serious Applications (VS-Games), Vienna, Austria, 2019, pp. 1-8.

[21] Ribeiro, Matheus & Tori, Allan & Tori, Romero & Nunes, Fátima. (2019). Immersive game for dental anesthesia training with haptic feedback. 1-2. 10.1145/3306214.3338592.

[22] Ushaw, Gary & Sharp, Craig & Hugill, Jessica & Rafiq, Sheima & Black, Carla & Casanova, Therese & Vancleef, Kathleen & Read, Jenny & Morgan, Graham. (2017). Analysis of Soft Data for Mass Provision of Stereoacuity Testing Through a Serious Game for Health. 216-220. 10.1145/3079452.3079496.

[23] Kaul, Paras. (2006). Neurological gaming environments. 25. 10.1145/1179295.1179321.

[24] Gazihan Alankus, Amanda Lazar, Matt May, and Caitlin Kelleher. 2010. Towards customizable games for stroke rehabilitation. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10). ACM, New York, NY, USA, 2113-2122.

[25] K. Choi, "Virtual Reality Wound Care Training for Clinical Nursing Education: An Initial User Study," 2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR), Osaka, Japan, 2019,

pp. 882-883.

[26] Bozkurt, Aras. (2017). Gamification, Education and E-learning: An interview with Andrzej Marczewski.

[27] Pastushenko, Olena & Geurts, Luc & Hruska, Tomas. (2019). Conceptual Learning of Electric and Electronic Circuits With Gamification. 589-596. 10.1145/3341215.3356305.

[28] Coull, Natalie & Donald, Iain & Ferguson, Ian & Keane, Eamonn & Mitchell, Thomas & Smith, Oliver & Stevenson, Erin & Tomkins, Paddy. (2017). The Gamification of Cybersecurity Training. 108-111. 10.1007/978-3-319-65849-0_13.

[29] Ard Jacobs, Annick Timmermans, Marc Michielsen, Maaiken Vander Plaetse, and Panos Markopoulos. 2013. CONTRAST: gamification of arm-hand training for stroke survivors. In CHI '13 Extended Abstracts on Human Factors in Computing Systems (CHI EA '13). ACM, New York, NY, USA, 415-420. DOI: 10.1145/2468356.2468430

[30] Diehl LA, Souza RM, Gordan PA, Esteves RZ, Coelho ICM. InsuOnline, an Electronic Game for Medical Education on Insulin Therapy: A Randomized Controlled Trial With Primary Care Physicians. J Med Internet Res 2017 Dec 09;19(3):e72 doi: 10.2196/jmir.6944

[31] Gauthier A, Corrin M, Jenkinson J. Exploring the influence of game design on learning and voluntary use in an online vascular anatomy study aid. Computers & Education 2015 Sep;87:24-34. doi: 10.1016/j.compedu.2015.03.017

[32] Boada I, Rodriguez-Benitez A, Garcia-Gonzalez JM, Olivet J, Carreras V, Sbert M. Using a serious game to complement CPR instruction in a nurse faculty. Comput Methods Programs Biomed 2015 Nov;122(2):282-291. doi: 10.1016/j.cmpb.2015.08.006

[33] https://www.nhs.uk/conditions/stroke/

[34] http://www.nihstrokescale.org/

Scalable Spherical Harmonics Hierarchies

Jonathan B. Metzgar
University of Alaska Fairbanks
Department of Computer Science
PO Box 756670
Fairbanks, Alaska, USA, 99775-6670
jmetzgar@acm.org

Sudhanshu K. Semwal
University of Colorado Colorado Springs
Department of Computer Science
1420 Austin Bluffs Pkwy
Colorado Springs, Colorado, USA 80918
semwal@uccs.edu

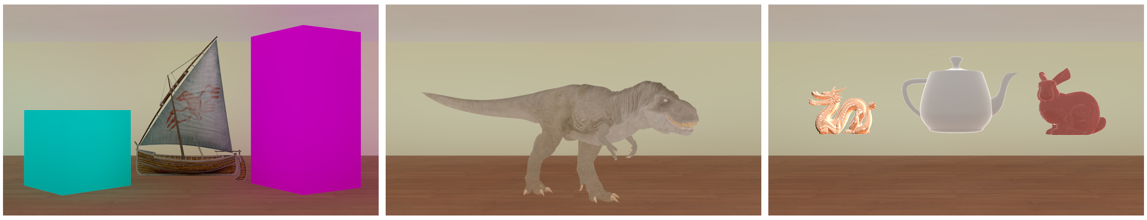


Figure 1: Scalable Spherical Harmonics Hierarchies used to generate global illumination.

ABSTRACT

Scalable Spherical Harmonics Hierarchies (SSPHH) is a real-time rendering solution to the global illumination problem. Our novel method is a system of components that enables the computation of light probes and the conversion to spherical harmonics coefficients, which are used as anisotropic time-varying light sources we call Spherical Harmonics Lights (SPHLs). The SPHLs encode irradiance information that can be used for image-based lighting. Our approach focuses on reconstructing scene lighting using diffuse illumination but is flexible to allow specular details. Furthermore, we consider the light transport from neighboring SPHLs by computing a transfer coefficient that estimates how much light from one probe is visible at another. SPHLs can be used for physically-based lighting using rendering methods similar to point lights and shadow maps. We created a reproducible testing methodology to compare our images with those of a commercial path tracer by automatically generating the appropriate scene information which gets used with absolute error metrics to determine remaining image defects. Our SSPHH method utilizes a scalable architecture to distribute the rendering of light probes between client and worker nodes using the ZeroMQ Majordomo protocol.

Keywords

spherical harmonics, anisotropic point lights, physically based rendering, path tracing, local dynamic radiance maps

1 INTRODUCTION

Real-time global illumination is a difficult problem to solve and understand for practitioners with numerous solutions proposed over the last several decades. Although new hardware is posing to change the focus to real-time path tracing, approximation methods will still have relevance for years to come due to problem complexity, power, space, and cost factors. Furthermore, it is difficult to offer solutions that solve the problem generally. Our work is a novel way of simulating global illumination using a network of light probes represented by spherical harmonics that estimates the transport of light from one node in the network to another. But let us begin by way of an illustration.

Imagine for a moment a house on a sunny day with several rooms, all with the doors closed and heavy curtains drawn. If you were to open the curtains in room A, photons immediately begin flooding the house and spreading through every crack they can find. In computer graphics, this room would be lit, and probably no light would leak through the cracks. Let us consider room B which has no direct illumination from room A, but now the door is open. If you were to open the door to room A, even though no direct light can reach B, indirect light is scattered from room A to room B. A path tracer would likely scatter photons to room B, but a typical real-time renderer using shadow maps renders no light in the second room. One typical solution for the real-time renderer uses baked global illumination, but baking forgoes the use of dynamic changes in scenery. So what can we do?

Our method supposes that you could configure light probes in each of the rooms and hallways and a set of edges connecting them. Then we will determine a visibility or transfer factor between neighboring vertices that would determine the amount of light that scatters from one light probe location to another. We use spherical harmonics to represent the visibility factor and the light probes. We use a four-part method of initialization, light probe visibility determination, light probe generation, and hierarchical spherical harmonic light generation. These four parts are components of our algorithm *Scalable Spherical Harmonics Hierarchies* (SSPHH) [Met18][1]. This paper is organized by reviewing previous research in this area before we talk about our theoretical framework, implementation, and results.

2 PREVIOUS WORK

Spherical harmonics (SH or SPH) are often used to represent low-frequency light probes because they act as a low pass filter and have some useful convolution and summation properties. In computer graphics, they represent the angular distribution of light. They are the spherical analog to the Fourier series and are a set of orthogonal functions. The spherical harmonics equation [Mer98] is given by

$$Y_m^\ell(\theta,\varphi) = K_m^\ell \, P_m^\ell(\cos\theta)\, e^{im\varphi} \,,$$

where

$$K_m^\ell = (-1)^m \sqrt{\frac{2\ell+1}{4\pi}\frac{(l-m)!}{(l+m)!}} \,,$$

$P_\ell^m(\cos\theta)$ is the associated Legendre polynomials, and $e^{im\varphi}$ can be represented in real form by

$$|e^i m\varphi| = \begin{cases} \cos m\varphi & \text{if } 0 \le m \le \ell \\ \sin m\varphi & \text{if } -\ell \le m < 0 \end{cases} .$$

The spherical harmonics are orthogonal polynomials which form an orthogonal set. This means that they satisfy the property that

$$\int_{\theta=0}^{\pi}\int_{\varphi=0}^{2\pi} Y_\ell^m(\theta,\varphi)\, Y_{\ell'}^{m'}(\theta,\varphi)\, \sin\theta \, d\varphi \, d\theta = \delta_{\ell\ell'}\,\delta_{mm'} \,,$$

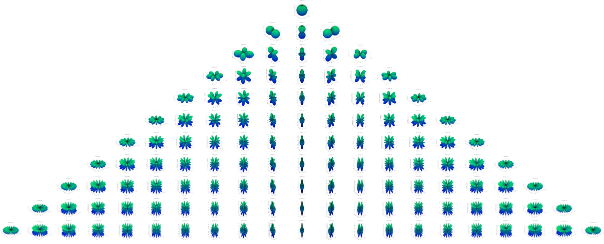


Figure 2: The degrees and bands of the spherical harmonics from $0 \le \ell \le 10$.

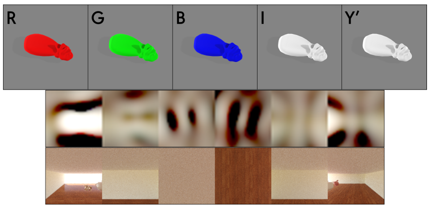


Figure 3: A SH representation of a light probe cube map.

where δ_{ij} is the Kronecker Delta

$$\delta_{ij} = \begin{cases} 0 & \text{if } i \ne j, \\ 1 & \text{if } i = j. \end{cases} .$$

We may now approximate a spherical function $L(\theta,\varphi)$ by calculating coefficients a_ℓ^m for each degree ℓ and band m with the integral

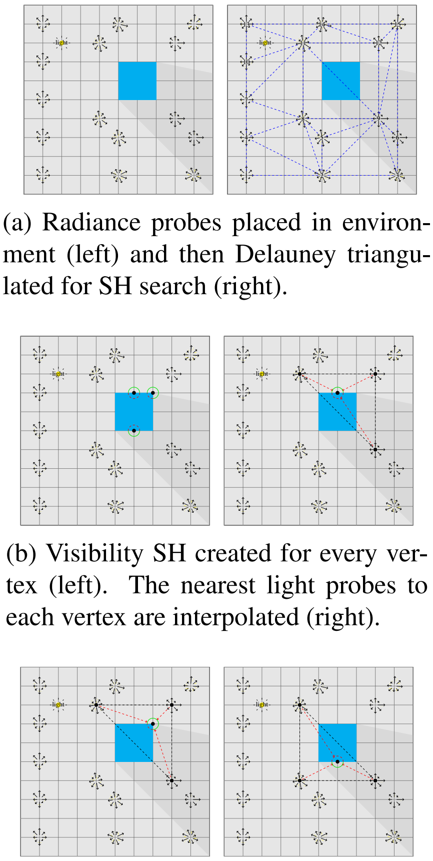
$$a_\ell^m = \int_{\theta=0}^{\pi}\int_{\varphi=0}^{2\pi} L(\theta,\varphi)\, Y_\ell^m(\theta,\varphi)\, \sin\theta \, d\varphi \, d\theta \,,$$

and we can reconstruct the original function by summing the spherical harmonics for $0 \le \ell \le N$ with the equation

$$L'(\theta,\varphi) = \sum_{\ell=0}^{N}\sum_{m=-\ell}^{\ell} a_\ell^m \, Y_\ell^m(\theta,\varphi) .$$

We use the `orthopoly` package from Maxima [S+82] to calculate the equations for $0 \le \ell \le 10$ and optimized those equations for calculation in our mathematics library. The number of SH coefficients is $(\ell+1)^2$ so we limit the highest degree to 10 since which require 121 coefficients. Figure 2 shows degrees $0 \le \ell \le 10$ and Figure 3 shows an example of a light probe which has been converted to a SH using the sampling approach from Green [Gre03]. In this paper, we use the operator $\bigotimes$ to represent *convolution* and operator $\bigoplus$ to represent *summation* which are component wise multiplication or addition of vector components. These are common symbols used in digital signal processing diagrams. We want to avoid confusion with tensor products and set theory.

[1] We abbreviated our algorithm SSPHH before realizing that SPH could be confused with Smoothed Particle Hydrodynamics, so we use SH for spherical harmonics and continue to use SSPHH for our algorithm.



(a) Radiance probes placed in environment (left) and then Delauney triangulated for SH search (right).

(b) Visibility SH created for every vertex (left). The nearest light probes to each vertex are interpolated (right).

(c) Two examples show algorithm using spherical harmonics based on visibility.

Figure 4: Demonstration scene showing Irradiance Volumes and Precomputed Radiance Transfer (IR/PRT) algorithm.

The use of the terms *radiance probes*, *irradiance volumes*, *environment maps*, and *light probes* refer to the same basic idea. The simple explanation is that they are multispectral 360° photographs taken at a point in space. The most common representation of image elements are high dynamic range RGB pixels, but they can be any single or multispectral representations. Irradiance volumes may be used to describe the hemisphere or spherical angular distribution of light around a point on a surface. A surface point may be shaded by using interpolated surrounding light probes to estimate irradiance. Environment maps are used for simulating reflection or irradiance. Lastly, any of these light probe images may be projected into spherical harmonics.

2.1 Path Tracing

Path tracing [Kaj86] and ray tracing [Whi80] are general solutions to the problem. Newer classes of hardware graphics processing units such as the NVIDIA GTX 2080 Ti are capable of ray tracing a few rays per pixel and couple it with image denoising [MMBJ17] to reconstruct indirect illumination. As path tracing enjoys the most recent industry support, we employ it in our method to construct light probes using Corona Renderer [OK16]. These methods remain impractical for the foreseeable future for use with real-time graphics when dealing with low energy or low compute devices. Our method falls into the category of a hybrid approach which will be a popular approach for some time.

Kajiya's rendering equation [Kaj86]

$$\mathbf{L}_o(\mathbf{x} \rightarrow \omega_o) = \mathbf{L}_e(\mathbf{x} \rightarrow \omega_o) + \int_{\Omega} f_r(\omega_i, \omega_o)\, \mathbf{L}_i(\omega_i \rightarrow \mathbf{x})\, \langle \omega_i, \omega_o \rangle\, d\omega_i \,.$$

is often solved with Monte Carlo integration, but we may eliminate the integral and process specific paths of light with the Dirac-delta function

$$\delta(x) = \begin{cases} +\infty, & x = 0 \\ 0, & x \neq 0 \end{cases},$$

where $\int_{-\infty}^{\infty} \delta(x)\, dx = 1$ and the integral of $f(x)$ which is only non-zero at $x = 0$ may be directly calculated

$$f(0) = \int_{-\infty}^{\infty} f(x)\, \delta(x)\, dx = \int_{-\infty}^{\infty} f(x)\, dx \,.$$

2.2 Precomputed Radiance Transfer

Irradiance Volumes [GSHG98, RH01] and Precomputed Radiance Transfer [SKS02] (IR/PRT) modify the rendering equation by utilizing transfer functions. The first transfer function determines self-shadowing or visibility from any direction. The second transfer function determines how much light illuminates the surface. Mathematically, this is the following operation

$$T(\mathbf{x} \rightarrow \omega_o) = \frac{1}{\pi} \int_{\Omega} L_{SH,\omega_i} \otimes H_{SH,\omega_i}\, d\omega_i$$

where $T(\mathbf{x} \rightarrow \omega_o)$ is the exit radiance at the point $\mathbf{x}$, L_{SH,ω_i} is the incident radiation at point $\mathbf{x}$ and H_{SH,ω_i} is the cosine weighted kernel which maps visibility from angle ω_i to point $\mathbf{x}$. This particular example highlights diffuse reflection and more sophisticated examples exist for specular and refractive materials.

Figure 4 shows the process of precomputed radiance transfer using irradiance volumes. Irradiance volumes are sampled as a set of SH coefficients. Each vertex on the world geometry gets a *transfer vector* that measures the angular light that is received at infinity. The shape of the SH looks like a measurement of the visible hemisphere; hence, the transfer vector has values close to 0 for directions opposite to the normal and values close to 1 near the normal. The IV probes are placed manually or procedurally in such a way that a Delaunay triangulation can be created. Every visible vertex will be contained in a simplex of this triangulation, or extrapolation could be used otherwise. The closest IV probes are interpolated using barycentric coordinates to estimate the illumination for that vertex. Lower performing hardware often use second order spherical harmonics (9 floats per RGB channel) or order four or five spherical harmonics (25 or 36 floats per RGB channel). The irradiance volumes may be produced dynamically, but *baking* eliminates the real-time performance hit and enables the use of higher quality path traced light probes.

2.3 Virtual Point Lights

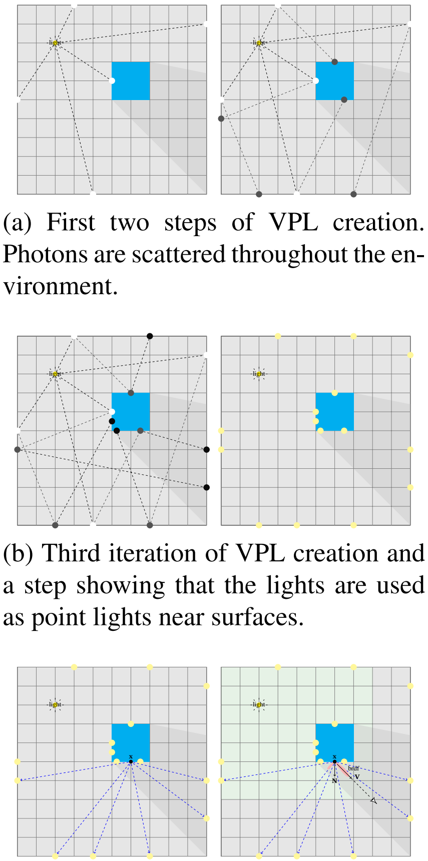
Instant radiosity—more generically *many light methods*—are used to simulate global illumination in a 3D scene by simulating a small set of "photons" and then lighting the visible scene with those photons. Ritschel et. al. [RDGK12] survey a wide class of global illumination algorithms of this sort. Typically, point lights or spherical lights [HKWB09] are used in these applications. Hot spots or fireflies can plague this class of algorithms because of discontinuities due to the nature of point lights especially if they are close to surfaces due to the $1/r^2$ inverse ratio, so clamping is used to control this problem.

Figure 5 shows the process of virtual point lights. Virtual point light locations are determined by scattering photons into the 3D geometry. These point lights are then used to simulate the indirect illumination of the scene using shadow mapping. One key problem with VPLs are hot spots near the location of the VPLs due to the quadratic falloff of the point lights. To partially correct this defect, virtual spherical lights can be used. To reduce the number of VPLs in a scene, *light cuts* may replace close groups of lights with a single VPL.

2.4 Voxel Based Methods

Voxel based methods using a 3D Digital Differential Analyzer (3DDDA) [FTI86] are also potential solutions to the global illumination problem. Voxel cone tracing [CNS$^+$11] is a recent method which projects lighting into a sparse octree and traces cones from surface geometry using final gathering. Other methods of 3DDDA use skip encoding to quickly march rays through the data structure [SK97]. Memory is a big problem with the voxel approach if you want high detail lighting. And light leakage may occur near geometry which does not conform to the voxel grid.

Figure 6 shows the process of voxel cone tracing. Rather than tracing rays, VCT proceeds by building a spatial subdivision data structure in which to propagate a light solution. The geometry is discretized into a voxel representation using a sparse octree. Afterwards the light is projected through the voxel grid to estimate irradiance in each cell of the octree which includes a filtering step to down-sample radiance. The camera pass then uses a normal direct illumination step and a secondary bounce step. The secondary bounces are traced through the octree using cone tracing which gathers indirect radiance.



(a) First two steps of VPL creation. Photons are scattered throughout the environment.

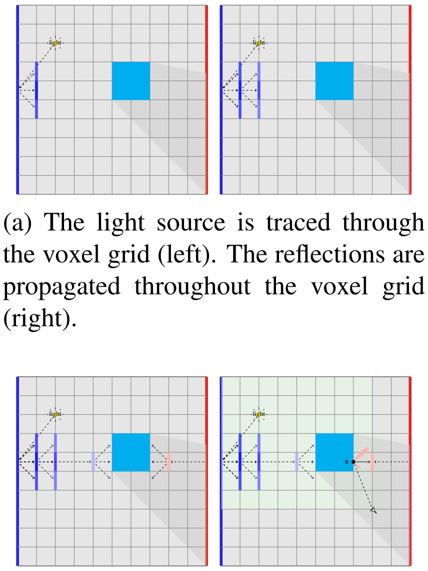
(b) Third iteration of VPL creation and a step showing that the lights are used as point lights near surfaces.

(c) Shadow maps determine visibility to **x** (left) and resulting light contribution (right).

Figure 5: Demonstration scene showing Virtual Point Lights (VPL) algorithm.

3 THEORETICAL FRAMEWORK

The SSPHH method solves the problem of global illumination using *anisotropic lights*, which we call *spherical harmonic lights* (SPHL). We begin by determining the SPHL locations, or *nodes*. We need to determine the amount of light scattered around an SPHL that reaches neighboring SPHL nodes by simulating a light source and measuring the light observed by the neighbor. The graph relationship of each node to another is called a *hierarchy*. We render the scene at each of the SPHL nodes and project the images to spherical harmonics. We construct the resulting SPHL coefficients by summing its own light probe with an adjusted version of the neighbors. Figure 7 shows a visual progression of our algorithm. In this section, we will discuss this idea from several perspectives.



(a) The light source is traced through the voxel grid (left). The reflections are propagated throughout the voxel grid (right).

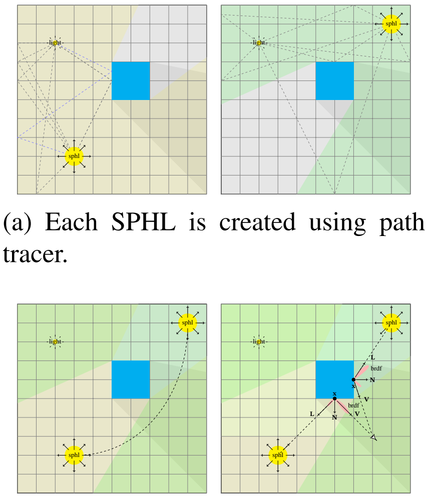
(b) Light transport between voxels eventually arrives at objects (left). During rendering, the voxels are used to shade surfaces (right).

Figure 6: Demonstration scene showing Voxel Cone Tracing (VCT) algorithm.

Let us return to the earlier house example. We see that if we place an SPHL in room A, place an SPHL in room B, and calculate the visibility SH coefficients from A to B, that room B will now have indirect illumination from room A without requiring a full path traced solution. This is because we can estimate how much light is transported from the SPHL in room A to the SPHL in room B. The remaining problem is time. If we slowly open the door to room A (or B), then we need to have a way of providing time-based interpolation of SH coefficients to accommodate the change in visibility between coefficients. We have not focused on this part of the problem in this paper.

If we consider our problem using regular expression notation [Hec90], we may use $L\{D|S\}E$ for direct illumination paths and $L\{D|S\}^{+}E$ for indirect illumination paths. The light path $L\{D|S\}^{+}E$ represents a chain of scattering events. We can divide this light path into three sections. The first is $L\{D|S\}^{*}\mathbf{S}_j$ which is the possible scattering event from a light source to the SPHL node $\mathbf{S}_j$. The second section is $\mathbf{S}_j\{D|S\}^{+}\mathbf{S}_i$ which is a chain of scattering events between SPHLs $\mathbf{S}_j$ and $\mathbf{S}_i$. Finally, we have the chain of scattering events $\mathbf{S}_i\{D|S\}\mathbf{x}$ which is the scattering event from SPHL $\mathbf{S}_i$ to the point $\mathbf{x}$ we are lighting.

In our method, we handle direct light paths $L\{D|S\}\mathbf{x}$ from various sources of lights including environment maps, directional lights, point lights, irradiance volumes, and SPHLs. The latter can be used to either simulate anisotropic time-varying light sources [MS12], or



(a) Each SPHL is created using path tracer.

(b) Light transport is simulated between SPHLs before determining indirect illumination in final render.

Figure 7: Demo scene showing SSPHH algorithm.

light probes to perform image based lighting. Later we call attention to this application, but for now we continue the discussion on light probes.

We can use an SPHL $\mathbf{S}_i$ to estimate the irradiance at point $\mathbf{x}$ using the radiance function $\mathbf{L}_{\mathbf{S}_i}(\theta,\varphi)$ where (θ,φ) is the unit length direction $\mathbf{L}_{\mathbf{S}_i}$ from $\mathbf{x}$ to $\mathbf{S}_i$. The vector (θ,φ) is used as the input for the spherical harmonic $Y_m^{\ell}(\theta,\varphi)$. The limitation is that we need line of sight from $\mathbf{S}_i$ to $\mathbf{x}$ but we have a workaround for this which we discuss shortly. Since diffuse illumination of a rough surface is uniformly scattered by the BRDF $f_r(\omega_i,\omega_o)=\rho$ at point $\mathbf{x}$, the value

$$\mathbf{I}=\frac{\mathbf{L}_{\mathbf{S}_i}(\theta,\varphi)}{\mathbf{N}_{\mathbf{x}}\cdot\mathbf{L}_{\mathbf{S}_i}}$$

is the irradiance of $\mathbf{x}$ which accounts for the $\langle\omega_i,\omega_o\rangle$ term in the rendering equation integral.

We use a matrix $\mathbf{H}$ to store the SPHL coefficients of the SSPHH hierarchy and a matrix $\mathbf{P}$ to store the transfer coefficients $\mathbf{P}_{ij}$ between SPHLs. The SH $\mathbf{H}_{i\to i}$ is the SPHL generated by the GEN step. The SH $\mathbf{H}_{j\to i}, i\neq j$ is created by the VIZ step. The GEN light probe $\mathbf{S}_i$ is composed of the light paths $L\{D|S\}^{*}\mathbf{S}_i$. The visibility coefficients $\mathbf{H}_{j\to i}$ is composed of the light paths $\hat{L}_{1,j}\{D\}^{*}\mathbf{S}_i$ where $\hat{L}_{1,j}$ is a spherical light source of total intensity 1. We have chosen the name $\mathbf{P}_{ij}$ because we believe this represents the probability that light from direction (θ,φ) at SPHL $\mathbf{S}_j$ will appear from direction (θ,φ) at SPHL $\mathbf{S}_i$; hence, $\mathbf{P}_{ii}=1$. So by convolving the SH coefficients with the equation $\mathbf{H}_{j\to i}\otimes\mathbf{S}_j$ and summing we get the resulting SPHL

$$\mathbf{S}'_i=\mathbf{S}_i\oplus\sum_{j=0,j\neq i}^{N-1}\mathbf{H}_{j\to i}\otimes\mathbf{S}_j$$

Algorithm 1 Scalable Spherical Harmonics Hierarchies Algorithm

```
function INIT
    SPHLs ← NEWSH(0)
end function

function VIZ
    for $i \leftarrow 0, \dim \mathbf{S}$ do
        for $j \leftarrow 0, \dim \mathbf{S}, i \neq j$ do
            $\mathbf{H}_{i,j} \leftarrow$ RENDERVIZCUBE($S_i$, $S_j$)
            $\mathbf{P}_{i,j} \leftarrow$ COMPUTEVIZTRANSFER($\mathbf{H}_{i,j}$)
        end for
    end for
end function

function GEN
    for all $\mathbf{S}_i \in$ SPHLs do
        $\mathbf{S}_{i,\text{lightprobe}} \leftarrow$ RENDERCUBEMAP($\mathbf{S}_i$)
        $\mathbf{S}_i \leftarrow$ CUBEMAPTOSH($\mathbf{S}_{i,\text{lightprobe}}$)
        $\mathbf{H}_{i,i} \leftarrow \mathbf{S}_i$
        $\mathbf{P}_{i,i} \leftarrow 1$
    end for
end function

function HIER($N$)
    for $i \leftarrow 0, N'$ do
        Create sequence of pairs
        Pairs are indexes and visibilities
        $Q = \{(q, p) \in \{0, 1, \ldots, N-1\} \times \mathbf{P}_i\}$
        Sort the previous sequence $Q$ by visibility
        $Q' = \{(q_j, p_j) \in Q | p_j > p_{j+1}, 0 \leq j < N\}$
        $\mathbf{S}'_{i,\text{self}} \leftarrow \mathbf{S}_i$
        Accumulate top $N'$ spherical harmonics
        $N' \leftarrow \min(N, \dim \mathbf{S})$
        $\mathbf{S}'_{i,\text{neighbor}} \leftarrow \sum_{j=0, j\neq i}^{N'} \mathbf{H}_{i,q_j} \otimes \mathbf{S}_{q_j}$
        $\mathbf{S}'_i \leftarrow \mathbf{S}'_{i,\text{self}} \oplus \mathbf{S}'_{i,\text{neighbor}}$
    end for
end function
```

where N is the number of SPHLs. This simulation of light transport from one SPHL to another is called the HIER step. We use the term hierarchy because we may sort the SPHLs by $\mathbf{P}_{ij}$ which rank the SPHLs by transport probability.

Now, this may not be not a scalable approach because we need an $N \times N$ sized matrix. We instead allow for a sparse matrix approach where the connections can be optimized to reduce memory and computational complexity. This is accomplished with a hash map instead of an array if the connection graph is not a complete graph. We create a list $\mathbf{Q}$ which contains pairs (q_i, p_i) of SH coefficients q_i and the transfer coefficients p_i which can be sorted based on p_i. We choose N' neighboring SPHLs instead and use p_i as a heuristic. If p_i is below a certain threshold, or we choose N' to be a specific number, then our method is *scalable*. So if $p_i = 1$, as all GEN nodes are, then those are included in the sum, and if $p_i > 0$, as all enabled VIZ nodes are, then our resulting SPHL is

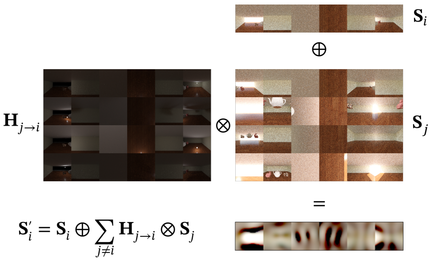

$\mathbf{S}_i$
$\oplus$
$\mathbf{H}_{j\to i}$
$\otimes$
$\mathbf{S}_j$
$=$
$\mathbf{S}'_i = \mathbf{S}_i \oplus \sum_{j\neq i} \mathbf{H}_{j\to i} \otimes \mathbf{S}_j$

Figure 8: Generating an SPHL $\mathbf{S}'_i$ with GEN and neighbor VIZ probes.

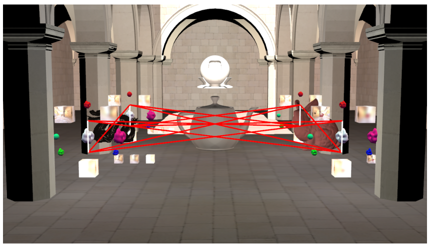


Figure 9: Image shows hierarchies and SH previews.

$$\mathbf{S}'_i = \mathbf{S}'_i \oplus \sum_{j=0, j\neq i}^{N'-1} \mathbf{H}_{q_j} \otimes \mathbf{S}_j \,.$$

Figure 8 shows a visual calculation of $\mathbf{S}'_i$ and Figure 9 shows an example set of hierarchies. Finally Algorithm 1 shows the complete four step process of the SSPHH algorithm.

4 IMPLEMENTATION DETAILS

Our research graphics engine *Fluxions* was designed for this algorithm and to work on a number of other graphics problems. There are several components worthy of mention. Our *Unicornfish* sub-library is an implementation of the majordomo protocol [Hin14] which is used to coordinate a number of worker nodes using a reliable request-reply pattern to help offload rendering tasks to other computing nodes in a local area network. We use it to send SH coefficients back to the main program using the JSON schema discussed in [MS19]. JSON schemata can be very helpful in client-server applications by adding validation to data interchange. JSON provides a blend of convenience for editing and readability. Figure 10 shows the process of converting a cube map with Corona to a SH representation which gets sent in JSON format for the renderer to use.

Processing HDR light probes require attention to data layout especially with cube maps. In practice, there is no consistency with how this data will be stored. The

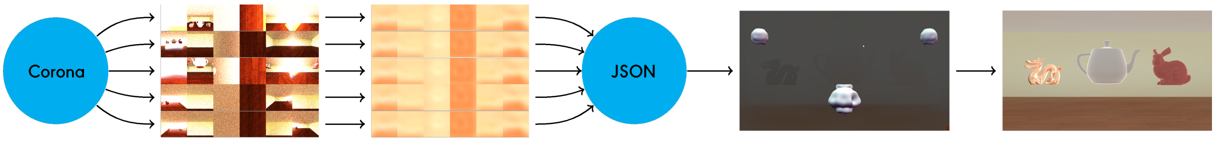


Figure 10: We outsource the light probe render to Corona, project to SH, and transmit JSON over a network.

most useful way to use a single rectangular image—rather than six individual images—is to store them in the order $(+X,-X,+Y,-Y,+Z,-Z)$ which is expected with OpenGL and Direct3D. We will mention that common layouts include the 6×1 horizontal strip, 1×6 vertical strip, 3×4 horizontal cross, or 4×3 vertical cross. The strip approaches have the benefit of not wasting image space.

Once we decided the best data formats to use in our SSPHH algorithm, we worked on a data collection plan to measure quality. There is no general consensus about how to measure image quality. No two renderers will be able to produce identical images in today's market. This is primarily due to the fact that every renderer is effectively a combination of several algorithms chosen for their application to a particular problem set. Even non-biased path tracers can generate different results if the authors chose to implement different BRDFs in their algorithms. What is best at this point is to use sound methodology in measuring image quality.

We chose to use Corona Renderer due to its usage of physically based rendering algorithms. The renderer needed to generate light probes and produce high quality images quickly—if any physically based path tracer can be described as *quick!* We also use Corona Renderer to generate ground truth images. We made some non-obvious choices about features to include in our rendered images. For example, we do not use bump maps in the reference images because not every renderer interprets bump maps equivalently. We believe it is better to use scene geometry which use the least amount of interpretable details as possible. So we limit our material exports to Lambertian diffuse surfaces, an optional specular roughness parameter, and texture maps. The reason for allowing specular roughness to be optional is to compare what highly varying BRDFs might do to an indirect illumination process.

Our testing program allows us to generate an identical ground truth image frame which we calculate *absolute error* to determine image differences. Because absolute error (or even RMSE) can have problems near polygonal edge boundaries, we also produce a low resolution image where we sum all the pixel data in smaller portions of the image. We do not take the average because we want to compare absolute differences. This approach enables us to visualize the total energy difference from reference and test images which can be see in Figure 13. This *pixelated* approach is used to help us understand where we get the most deviation. However, unlike normal pixelation, we sum up the energies in that area of the image rather than take the mean. We do not include AE numbers in our paper, but we have used them to focus our attention to why certain image areas have larger energy differences and we discuss this later in the analysis section.

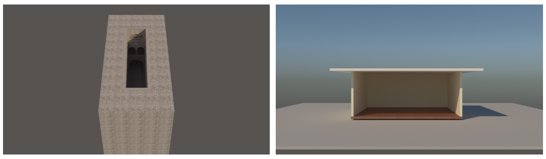


Figure 11: The two test scenes environments showing the requirement for indirect illumination.

The implementation of spherical harmonic lights is similar to point lights and point light shadow maps. A shadow map is dynamically generated for each SPHL node. The shadow map is used to determine whether to use the GI contribution. Since the SPHL stores the irradiance, only one SPHL needs to be used. If no SPHLs are available, then the closest one can be used to estimate the irradiance by assuming that the light probe is located around $\mathbf{x}$. In other words, instead of using the SPHL as a source of light, we use it as an irradiance volume instead. This is the case for surfaces where shadow mapping would cause significant amounts of occlusion. Please note that if you are using PRT or irradiance volumes, then you would want to interpolate the irradiance volumes that surround the vertex or shading fragment. The low frequency nature of spherical harmonics makes this a minor concern, and we save a deep analysis on defects arising from this usage for future work.

5 RESULTS AND ANALYSIS

Figure 1 shows two scenes with different objects. The first scene is designed to test color bleeding and the second scene is designed to test complex objects. Figure 13 show the comparison of the Corona renderer to our method. We also compare a ground truth image with a maximum ray depth of 3 since that represents second and third bounces of light. The reader will notice that there is a loss of energy (darkening) from a max ray depth of 25 in the ground truth images.

Table 1 shows the times needed to perform the Reference (REF), SPHL generation (GEN), SPHL visibility (VIZ), and hierarchical SPHL (HIER) steps. We compare both the time it takes to create reference quality

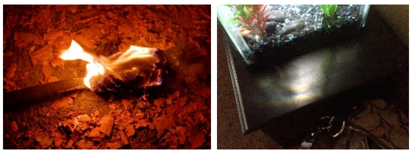


Figure 12: An SPHL could be used to represent complex emission profiles such as fire or caustics.

light probes—max ray depth 25 and pass limit 25—with normal quality light probes—max ray depth 3 and pass limit 1. For a set of four to six SPHLs with a resolution of $64 \times 64 \times 6$, less than one minute is required. This is generated on the same CPU, but in a different thread than the rendering thread. The minute time frame is due to the CPU based Corona Renderer which must load the geometry and materials and render an image. We are addressing this shortcoming by working on an internal GPU based path tracer to avoid the extra cost which we believe this can be done at 60Hz using 12 samples per pixel and a GPU based ray tracing framework.

We used our Majordomo protocol sub-library Unicornfish to launch Corona. This process converts the scene graph into the XML data format which Corona can read. Because spherical harmonics are a type of data compression, we are able to render the light probes on a separate rendering node and send them back efficiently. The size of those packets depend on the maximum degree of the SH to encode. For example, a tenth degree SH with RGB components is $3 \times (10+1)^2 = 363$. Formatting this in JSON requires under 4KB of formatted text with 5 digit accuracy. A binary form would require significantly less space. However, SPHLs do not need to be updated unless there is a significant change in the lighting context such as a door opening or closing.

We believe these would be very useful in high performance games. A secondary use would be to have a game client generate an SPHL for their own players and share that with a game server to distribute to neighboring players. SPHLs do not need to only represent global illumination, they can also represent anisotropic lights. A space ship or hot rodded automobile with custom lights could be constructed as an SPHL by sampling the emission profile of the player where

$$\mathbf{L}_o(\mathbf{x} \rightarrow \omega_o) = \mathbf{L}_e(\mathbf{x} \rightarrow \omega_o) \, .$$

Sending these to nearby players would result in object to object light interactions. Using SPHLs to represent dynamic local illumination from static objects such as torches, magical effects, or even fish tanks would result in more realistic experiences. Figure 12 shows complex emission profiles from fire and caustics.

Path Tracer Configuration

The maximum ray depth (MRD), samples per pixel, and pass count (PC) are three of the biggest factors in quality and render time required for ray tracers. Maximum ray depth deals with how many bounces of light are allowed. Using the regular expression notation, $L\{D|S\}E$ is one bounce of light, $L\{D|S\}^2E$ is two bounces of light, and so on. To prevent exponential growth of calculation as in Whitted ray tracing (e.g. shadow rays, refraction rays, reflection rays), only one path is chosen for a single sample. This is determined using Russian roulette and importance sampling. A path tracer which utilizes this technique should see maximum ray depth scale linearly with each additional bounce. The samples are scaled by the BRDF and averaged to create the final pixel. Many samples per pixel must be used to obtain a low-noise image. Pass count allows the image to be constructed progressively. That is, rather than wait for the appropriate number of samples to be recorded before producing an image, we calculate a certain number of samples per pixel per pass and update the image when each pass is finished.

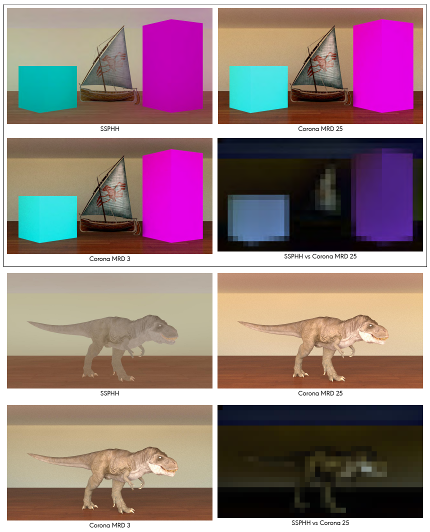


Figure 13: Additional test scenes showing color bleeding (top) and a complex model (bottom). Absolute error from reference images are also shown.

The second major factor that affects light probe render time is the quantity of geometry and light preprocessing before the image can be rendered. Intel's Embree library is a popular choice for accelerating ray-triangle intersections. And this is mostly a fixed cost, so there is not much that can be improved aside from choosing to precompute some of the lighting with a cache. Readers familiar with techniques like photon mapping [Jen96] will relate to the idea of scattering photons in a scene and reusing them to accelerate global illumination. Corona Renderer uses a similar technique called a UHD Cache which maintains similar quality with unbiased path tracing especially for indoor scenes. There is generally no benefit to turning the UHD Cache off and made no significant impact in the generation of

Table 1: Comparison of reference, GEN, VIZ, and HIER times for each scene.

Scene	*mrd*	*pl*	REF time (s)	GEN time (s)	VIZ time (s)	HIER time (s)
Gallery 1	25	25	164.46	131.54	N/A	0.62173
	3	1	15.03	48.40	197.46	0.91512
Gallery 2	25	25	176.10	141.39	N/A	0.62173
	3	1	18.35	53.08	230.81	0.61695
Gallery 3	25	25	161.43	141.76	N/A	0.60576
	3	1	16.37	56.25	234.89	0.69566
Sponza	25	25	220.20	151.12	N/A	0.47393
	3	1	21.88	55.60	178.34	0.48255

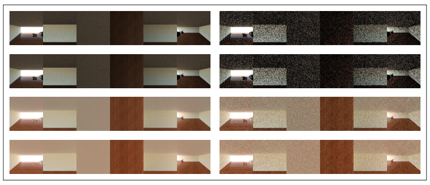


Figure 14: Varying maximum ray depth and pass limits for light probes. Left column uses a high pass count and the maximum ray depths from top to bottom are 0, 1, 3, and 25.

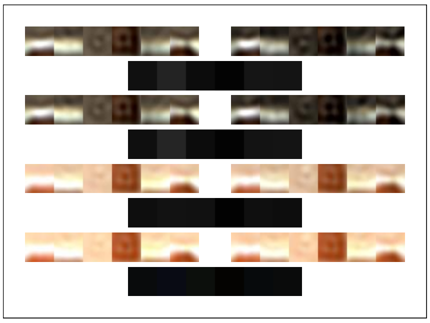


Figure 15: Varying maximum ray depth and pass limits for spherical harmonic lights. Left column uses a high pass count and the maximum ray depths from top to bottom are 0, 1, 3, and 25. The center column shows the difference between the left and right columns.

our light probe images. Instead, real-time performance was hindered by the resulting increased rendering time.

Adjusting the maximum ray depth and pass limit affects quality of the light probe images to varying degrees. Figure 14 shows the difference between path traced images using various MRD and PL. Figure 15 shows the effect of choosing different maximum ray depths and pass limits for creating SH. By comparing successive images, we determined that a practical depth of 3 provides enough light information while remaining sufficiently close in quality as a depth of 25.

Image Energy Differences

Matching images between two different renderers is challenging. We noted earlier that we use absolute error to mark differences between our images and the ground truth images as shown in Figure 13. Those deviations are primarily due to tone mapping, BRDF differences, and indirect illumination in our rendering system.

Indirect illumination in real-time renderers is often enhanced with ambient occlusion which estimates the accessibility of a surface point to ambient light. We do not simulate ambient occlusion in our images because this was out of scope and we wanted to focus on how well our algorithm worked on its own. So the reader will notice that the darkening of edges is not well handled in our approach. We expect that adding ambient occlusion will benefit a production image.

Secondly, we note that our reflection models do not match exactly with Corona Renderer. We utilize a bipolar version of normalized Blinn-Phong and GGX for specular reflection and a bipolar version of Oren-Nayar and Disney's BRDF [BS12]. We will not claim this is novel, but our bipolar approach means that the roughness parameter α maps from -1–1, where model A is chosen if $\alpha < 0$ and model B is chosen if $\alpha > 0$. This is different than Corona's BRDF which uses SGGX and Lambertian models. Our exporter simplifies the materials to a lowest common denominator of diffuse color, specular color, index of refraction, and roughness. We only support solid objects and thus do not handle transparency effects or subsurface scattering (e.g. the cloth on the the sailboat).

Lastly, tone mapping makes a huge difference in rendering and there are several approaches to make high dynamic range images fit inside the sRGB color space. We utilized gamma scaling coupled with an exposure control. Corona uses a similar adjustable gamma and exposure control along with filmic highlight compression. We chose to limit our tone mapping to something most likely to be used in real-time graphics, though we would like to experiment with more elaborate approaches. We did not modify the original tone mapping of the texture maps we loaded, while Corona does. This explains the darkening of the textures in the reference images versus ours. However, we perform tone mapping to render an sRGB image at the final stage. In the future, we are working on changing this approach to allow custom tone mapping properties on imported textures. Regardless, we always performed lighting calculations in linear space to match current best practice in physically based rendering.

6 CONCLUSION AND FUTURE WORK

In this paper, we presented the Scalable Spherical Harmonics Hierarchies method and how it can be used to approximate global illumination in real-time applications. We use spherical harmonic light probes to determine surface irradiance which allow us to simulate low frequency indirect illumination using local anisotropic point lights called SPHLs. We estimate the global illumination by determining the light transport from one SPHL to another. Hence, we can solve the problem of indirect light transport for solid objects in real-time by precomputing the visibility between SPHLs to determine how much light is transported.

In the future, we see this as a great application for client-server applications where global illumination information is dynamically updated for players in a video game context. On-line streaming platforms could utilize this concept to share global illumination between players in resource constrained environments. Lastly, we also want to investigate the use of our method for participating media and time-varying effects.

7 REFERENCES

[BS12] Brent Burley and Walt Disney Animation Studios. Physically-based shading at disney. *ACM SIGGRAPH*, 2012:1–7, 2012.

[CNS+11] Cyril Crassin, Fabrice Neyret, Miguel Sainz, Simon Green, and Elmar Eisemann. Interactive indirect illumination using voxel cone tracing. In *Computer Graphics Forum (Proc. of Pacific Graphics 2011)*, 2011.

[FTI86] A. Fujimoto, T. Tanaka, and K. Iwata. Arts: Accelerated ray-tracing system. *IEEE Computer Graphics and Applications*, 6(4):16–26, April 1986.

[Gre03] Robin Green. Spherical harmonic lighting: The gritty details. Technical report, Sony Computer Entertainment America, 2003.

[GSHG98] Gene Greger, Peter Shirley, Philip M. Hubbard, and Donald P. Greenberg. The irradiance volume. *IEEE Computer Graphics and Applications*, 18:32–43, 1998.

[Hec90] Paul S Heckbert. Adaptive radiosity textures for bidirectional ray tracing. *SIGGRAPH Comput. Graph.*, 24(4):145–154, 1990.

[Hin14] Pieter Hintjens. *ØMQ - The Guide*. iMatix Corporation, 2014.

[HKWB09] Miloš Hašan, Křivánek, Bruce Walter, and Kavita Bala. Virtual spherical lights for many-light rendering of glossy scenes. *ACM Trans. Graph.*, 28(5):143:1–143:6, 2009.

[Jen96] Henrik Wann Jensen. Global illumination using photon maps. In *Rendering Techniques 96*, pages 21–30. Springer, 1996.

[Kaj86] James T. Kajiya. The rendering equation. *SIGGRAPH Comput. Graph.*, 20(4):143–150, August 1986.

[Mer98] Eugen Merzbacher. *Quantum Mechanics*. Wiley, 3 edition, 1998.

[Met18] Jonathan B. Metzgar. *Realtime Physically Plausible Global Illumination Using Scalable Spherical Harmonics Hierarchies*. PhD thesis, University of Colorado Colorado Springs, 2018.

[MMBJ17] Michael Mara, Morgan McGuire, Benedikt Bitterli, and Wojciech Jarosz. An efficient denoising algorithm for global illumination. In *Proceedings of High Performance Graphics*, HPG '17, pages 3:1–3:7. ACM, 2017.

[MS12] Jonathan Brian Metzgar and Sudhanshu Kumar Semwal. Approximating the fire flicker effect using local dynamic radiance maps. In *International Conference in Central Europe on Computer Graphics and Visualization*. Václav Skala-UNION Agency, Pilsen, CZ, 2012.

[MS19] Jonathan B. Metzgar and Sudhanshu K. Semwal. Applying zeromq to realtime global illumination rendering. In *Symposium on Interactive 3D Graphics and Games (i3D) 2019*, 2019.

[OK16] Jaroslav Kivánek Ondej Karlík, Adam Hotový. Corona renderer. `https://corona-renderer.com/`, 2016.

[RDGK12] Tobias Ritschel, Carsten Dachsbacher, Thorsten Grosch, and Jan Kautz. The state of the art in interactive global illumination. *Comput. Graph. Forum*, 31(1):160–188, 2012.

[RH01] Ravi Ramamoorthi and Pat Hanrahan. An efficient representation for irradiance environment maps. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 497–500, New York, NY, USA, 2001. ACM.

[S+82] William Schelter et al. Maxima. `http://maxima.sourceforge.net/`, 1982.

[SK97] Sudhanshu K. Semwal and Hakan Kvarnstrom. Dual extent and directional safe zone techniques for ray tracing. In *Proceedings of Graphics Interface Conference, Kelowna*, BC, Canada, 1997.

[SKS02] Peter-Pike Sloan, Jan Kautz, and John Snyder. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Trans. Graph.*, 21(3):527–536, July 2002.

[Whi80] Turner Whitted. An improved illumination model for shaded display. *Commun. ACM*, 23(6):343349, June 1980.

Automated Microwave Tomography (MWT) Image Segmentation: State-of-the-Art Implementation and Evaluation

Yuchong Zhang
Chalmers University of Technology
Gothenburg, Sweden
yuchong@chalmers.se

Yong Ma
Ludwig Maximilian University of Munich
Munich, Germany
yong.ma@ifi.lmu.de

Adel Omrani
Karlsruhe Institute of Technology
Karlsruhe, Germany
adel.hamzekalaei@kit.edu

Rahul Yadav
University of Eastern Finland
Kuopio, Finland
rahuly@uef.fi

Morten Fjeld
Chalmers University of Technology
Gothenburg, Sweden
fjeld@chalmers.se

Marco Fratarcangeli
Chalmers University of Technology
Gothenburg, Sweden
marcof@chalmers.se

ABSTRACT

Inspired by the high performance in image-based medical analysis, this paper explores the use of advanced segmentation techniques for industrial Microwave Tomography (MWT). Our context is the visual analysis of moisture levels in porous foams undergoing microwave drying. We propose an automatic segmentation technique—MWT Segmentation based on *K*-means (MWTS-KM) and demonstrate its efficiency and accuracy for industrial use. MWTS-KM consists of three stages: image augmentation, grayscale conversion, and *K*-means implementation. To estimate the performance of this technique, we empirically benchmark its efficiency and accuracy against two well-established alternatives: Otsu and *K*-means. To elicit performance data, three metrics (Jaccard index, Dice coefficient and false positive) are used. Our results indicate that MWTS-KM outperforms the well-established Otsu and *K*-means, both in visually observable and objectively quantitative evaluation.

Keywords
Image Segmentation, Otsu, *K*-means, Microwave Tomography.

1 INTRODUCTION

1.1 Image segmentation and MWT

Image segmentation is one of the most commonly used methods to classify the pixels of an image correctly in decision oriented applications. Due to its capability for distinguishing various features [1], it is able to divide an image into a number of discrete regions such that the pixels have high similarity within and high contrast between regions [2]. It has been well proven that segmentation is widely beneficial to a range of tomographic systems especially Computed Tomography (CT) [3, 4, 5, 6, 7]. For example, applications like segmenting different tissue types—e.g., lungs, kidneys, livers—in a CT image are reliable for accurate and precise diagnoses. There are many application areas for tomographic systems. For instance, while the adoption of image segmentation has been successful on a Microwave Tomography (MWT) system, this is mainly concerning medical usage. This paper focuses on leveraging state-of-the-art segmentation methods to MWT images for specific physical attribute detection in an industrial process setting.

MWT is a non-ionizing imaging technique that provides a quantitative image of the dielectric profile of the object of interest (OI) [8, 9]. In this paper, we focus on using MWT to observe a specialized industrial process—microwave drying for porous foams. An important physical parameter—moisture level of the foam—is measured throughout the whole drying process. The moisture level is regarded as the measure of a complete drying process. Depending on the imaging techniques used, a reconstructed tomographic images can display either the location of the moisture through the foam as an OI or a map of the dielectric properties of the moisture value both commonly referred to as

https://doi.org/10.24132/CSRN.2020.3001.15

MWT techniques [10, 11]. In our setup, the foam to be dried is measured and characterized by MWT sensors processed by a series of imaging techniques. After image reconstruction, the dielectric properties of the foam represents a moisture level, depicted in Figure 1, which is an example of reconstructed images using the MWT system. Hence, with the aid of mixing rule techniques [12], an ad hoc mapping is defined to obtain the moisture level of the foam.

After a drying process, some regions of the foam may not be completely dry. It is obvious that dry parts have less moisture, hence the amplitude (moisture or dielectric level) of these areas is lower than in the non-dry regions. The difference between dry and non-dry is represented by allocating different colors to each region (Figure 1) where the blue parts stand for good/low-moisture and the yellow parts stand for bad/high-moisture. Dark blue parts are drier than yellow areas. The colours become more yellow when there is more moisture. It is critical for operators and practitioner to precisely assess the low moisture areas of the material so as to gauge the success of a drying process.

There are some efficient MWT systems that produce high-quality images for industrial processes [11, 9]. In our drying process, the measurement of the foams is also performed by MWT, whose detailed design is beyond the scope of this paper. We address the segmentation for visualizing the blue regions representing low moisture. Thus, we propose an automatic segmentation method called MWT Segmentation based on K-means (MWTS-KM) for processing the images reconstructed by an MWT in our experiment. This algorithm is fully automatic and the user only has a verification role [1]. Next, we report on how we carried out experiments in practical settings to validate the proposed segmentation method, including how we successfully validated its robustness, performance and reliability.

1.2 Contributions

As the full version of our recently published work [13], this paper possesses two main outcomes. Firstly, we propose to visualize the moisture area, more precisely the low moisture area for porous foams using image segmentation methods on MWT images, on the premise that many segmentation methods have demonstrated satisfactory utilization in tomographic systems but not desirable success in the context of MWT. Secondly, a state-of-the-art MWTS-KM segmentation method is proposed to visualize the low moisture area of an image. As a novel integrated algorithm, it is compared with two alternative segmentation algorithms that have been used widely in related application areas.

The organization of this paper is as follows. The related works section briefly presents parts of the current research situation in segmentation methods used in tomographic images; mostly from the medical application areas. The proposed algorithm and another two common algorithms for comparison are presented in Section 3. Section 4 elaborates the basic experimental setup and the results including evaluation. Discussions and conclusions are given in Sections 5 and 6 respectively.

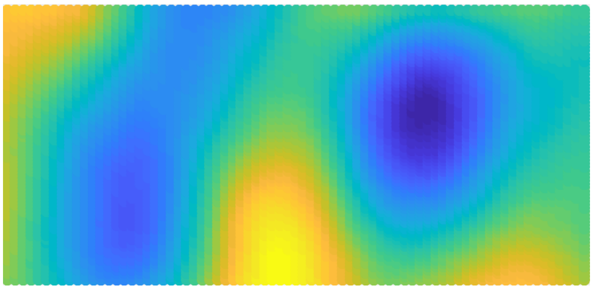


Figure 1: Example of an MWT image (input image) in our study. Blue means lower moisture, yellow means higher moisture.

2 RELATED WORK

Significant research efforts have been invested into the topic of tomographic image segmentation and related areas of application. Sharma et al. [14] provided a review of a set of automated segmentation methods ranked by applicability and suitability in the context of tomographic images; especially CT images. Shoaib et al. [15] used the Otsu algorithm [16] to propose a method including thresholding to prove a successful segmentation in lungs using CT. Likewise, Dorgham [1] deployed an automatic segmentation method on the basis of GrabCut [17] to detect human body Regions of Interest (RoI) from CT images. Wu et al. [18] also proposed an automatic segmentation algorithm named AUGC in ultrasound tomography images for breast cancer screening and pathological quantification. This method was also established on GrabCut as well, addressing incomplete labeling and speeding up multicore parallel programming. Moreover, Jose et al. [4] effectively detected and identified the exact location of a brain tumor through K-means clustering and fuzzy c-means algorithms by segmentation CT images. Sheppard et al. [19] utilized a combination algorithm containing active contours [20] and watershed transformation [21] to implement segmentation for porous materials in industrial tomographic images, resulting in superb quality results. Rashno et al. [22] developed a fully automated algorithm to segment fluid-filled and cyst regions in optical coherence tomography (OCT) retina images, segmented by combining a neutrosophic transformation and a graph-based shortest path method. Venhuizen et al. [23] developed a fully automatic segmentation system for segmenting retina in optical coherence tomography (OCT) through using a Convolutional Neural Network (CNN), proving reliable efficiency. Similarly, another variant end-to-end Deep Deconvolutional Neural Network (DDNN) was presented by Men et al. [6] for multi-target segmentation in nasopharyngeal cancer.

Concerning practical MWT, Wang et al. [7] employed a comparative study on MWT image segmentation for breast cancer detection. They used a *K*-Nearest Neighbor (KNN) algorithm and Gaussian Mixture Model (GMM) in their research, and showed that KNN outperformed GMM when segmenting Region of Interest (RoI) when using the Mathew Correlation Coefficient (MCC). Another concrete comparative study was presented by Mahmood et al. [5], who demonstrated that MWT image segmentation is even capable of ameliorating the accuracy of image reconstruction using several automated segmentation methods. Joseph et al. [24] integrated image segmentation with their inverse scattering algorithm called Forward- Backward Time-Stepping (FBTS) to apply on microwave imaging for brain tumors. From the retrospective work, we determine to deploy two well-known approaches as pre-study to support our methodology.

2.1 Otsu Algorithm

The Otsu algorithm is an unsupervised method which can be used to select a threshold automatically from a gray level histogram and deal with the problem of selecting the thresholds in images [16]. This method plays an important role in the image segmentation because of its advantages of simple implement and time performance [25] such as the documented image segmentation based on the Otsu method [26, 15] and infrared image segmentation based on the improved Otsu method [27]. It is a widespread methodology used in dividing images into two categories of pixels; foreground and background. In this paper, this method is employed to segment the MWT image into two subcategories according to different levels of moisture. The description of how to segment the MWT image using the Otsu algorithm is explained as follows:

1. Calculate the histogram and probability of each intensity level.
2. Set up the initial $\omega_i(0)$ and $u_i(0)$.
3. Traverse the threshold values from t=155 to 255, update ω_i and u_i, and compute the variance σ_b^2.

The segmentation threshold of the different moisture levels is denoted by t. The proportion of pixels belonging to a particular moisture level of the MWT image is recorded as $\omega_0(t)$, with the average gradation of $\mu_0(t)$. Moreover, the ratio of pixels in another moisture level is $\omega_1(t)$, and has an average gradation of $\mu_1(t)$. The average gradation of the entire MWT image is denoted by $\mu(t)$, the variance between classes is annotated by σ_b^2. Assuming the size of MWT image is M (width) * N (height), the number of pixels in which the gray value is lower than the threshold T in the corresponding image is N_0. The number of pixels in which the gray value is above the threshold T is N_1. Thus, we have:

$$\omega_0 = \frac{N_0}{M*N} \tag{1}$$

$$\omega_1 = \frac{N_1}{M*N} \tag{2}$$

$$N_0+N_1=M*N \tag{3}$$

$$\omega_0+\omega_1=1 \tag{4}$$

$$\mu=\omega_0*\mu_0+\omega_1*\mu_1 \tag{5}$$

$$\sigma_b^2=\omega_0(\mu_0-\mu)^2+\omega_1(\mu_1-\mu)^2 \tag{6}$$

$$\sigma_b^2=\omega_0\omega_0(\mu_0-\mu_1)^2 \tag{7}$$

The boundaries between the low moisture areas and other areas will be found due to the values in these two sets of pixels through MWT images.

2.2 *K*-means Algorithm

The *K*-means algorithm is a method of cluster analysis in data mining [28] which can also be used in image segmentation. This genre of unsupervised machine learning algorithm aims to minimize the sum of squared distances between all points and the cluster center [29, 30]. K denotes the number of centers we choose depending on the number of categories we intend to classify. In image segmentation, the *K*-means algorithm will help cluster images based on color and the minimum square Euclidean distance. In our study, this method is simultaneously used to find the boundary between the low moisture and other areas in MWT images. The value of K is chosen as two, as expected.

MWT pixel values range from 0 to 255; the size of the image is M (width) * N (height). Assuming that the matrix of pixels in this image is $[X_{11}, X_{12}, ..., X_{M-1,N}, X_{M,N}]$ while the value of K is two, the total pixel values in this matrix will be divided into two parts randomly. The first sets are $S_1=[W_1, W_2, ..., W_N]$ and the second sets are $S_2=[V_1, V_2, ..., V_M]$ separately. The overview and basic schematics of how the *K*-means method works in our study are elaborated as below:

1. Choose two samples randomly from the dataset D=[X_{11}, X_{12}, ..., $X_{M-1,N}$, $X_{M,N}$] as the initial centroid vector: u_1, u_2 and the first set is S_1 and the another set is S_2.

2. Calculate the distance between the element of dataset x_i from first set S_1 and second set S_2 and the centroid u_j: d_{ij}=$\|x_i$-$u_j\|^2$. If d_{i1}>d_{i2}, x_i will belong to S_1 otherwise x_i will belong to S_2.

3. Output the classification S=S_1, S_2.

The boundaries between different levels of moisture will be found according to the values in these two sets of pixels in the MWT image.

2.3 Performance Evaluation

The accuracy is chosen for gauging the performance in this paper. To evaluate the accuracy obtained from our segmentation algorithms, the similarity and false positive are measured by jointly applying three criteria: Jaccard index, Dice coefficient and false positive:

- *Jaccard index*: This is a commonly used metric to derive the similarity and diversity of finite sample sets. We choose this statistic to gauge our study as it has been extensively applied in the context of image segmentation [3, 18]. In our paper, it indicates the pixel-level similarity of input and output image. The input MWT image is referred as ground-truth while the output image represents the segmented image. The mathematical form of the Jaccard index is defined as (In this article, G and S annotate the input ground-truth image and the output segmented image respectively amid this denotation [18].):

$$Jaccard\ index = \frac{|G \cap S|}{|G \cup S|} \tag{8}$$

- *Dice coefficient*: This is another well-known parameter to measure the similarity between two samples. In our case, it will also result in pixel-level similarity between input and output. It differs from the Jaccard index which only calculate true positives once. The Dice coefficient normalizes the number of true positives to the average size of the two segmented areas [3]. It's defined as:

$$Dice\ coefficient = 2 \times \frac{|G| \cap |S|}{|G| + |S|} \tag{9}$$

- *False positive*: This is the expectancy of the false positive ratio. This rate is the ratio between the number of negative samples falsely classified as positive and the total number of negative samples. Low values indicate high accuracy. A false positive manifests in the form of:

$$False\ positive = \frac{|S| - |G \cap S|}{|G|} \tag{10}$$

The performance metrics yield values ranging from zero to one. Reaching higher Jaccard and Dice values close to one represents a desirable segmentation result with high accuracy while a value approaching to zero of false positives achieves nearly perfect segmentation performance.

2.4 Otsu VS *K*-means

The designed pre-study is to compare the functionality between Otsu and *K*-means. Hence, we implement the same segmentation principles by using these two renowned approaches with our total 30 MWT images. Due to the limited space and better interpretation, we randomly select four samples as presentation. The outcomes are shown in Figure 2. In addition, the metric-driven quantitative evaluation among Jaccard index, Dice coefficient, and false positive is illustrated in Figures 3-5. From the results obtained, it is noteworthy that the *K*-means evidently outperforms Otsu method in most samples. Therefore, this inspiration enables us to develop our proposed state-of-the-art algorithm.

3 METHOD

This section presents our proposed algorithm while Otsu and *K*-means algorithms are chosen for the comparative evaluation of the proposed algorithm in terms of segmentation performance because of their proven reliability and success over the past few years.

3.1 Overview of MWTS-KM

We have demonstrated that *K*-means exceeds Otsu in our context from the conventional methods' perspective. The *K*-means algorithm is the one of the simplest clustering algorithms and is computationally faster than the hierarchical clustering. It can also work for a large number of variables. [2]. As a type of unsupervised machine learning algorithm, it is widely used in the realm of image segmentation due to its simplicity and efficiency. Many researchers have tried to combine it with other techniques to increase its performance, such as integrating with fuzzy c-means algorithms [4] and partial stretching enhancement [2]. Following such a strategy of combination, our proposed algorithm—MWT Segmentation based on *K*-means (MWTS-KM)—is established based on the basic *K*-means algorithm. It integrates image augmentation, grayscale image conversion and conventional *K*-means into our proposed automatic algorithm. This integrative strategy of our method is

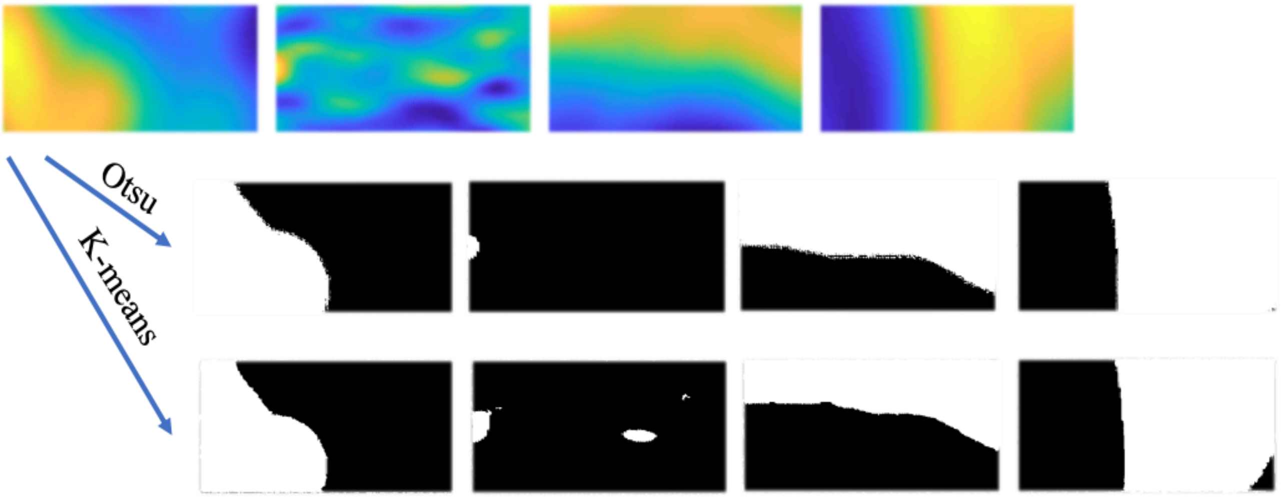


Figure 2: The comparison among Otsu and K-means. The first row shows the input images. The second row results from applying the Otsu algorithm on the input images while the third row results from K-means Algorithm.

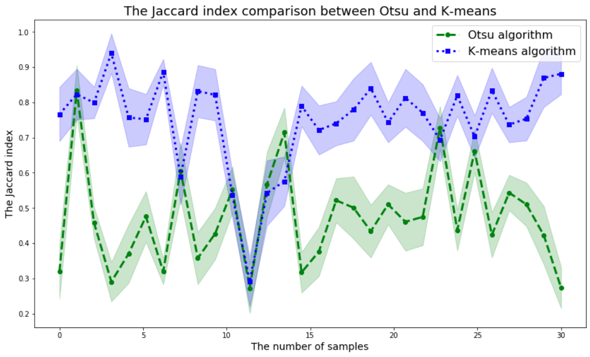


Figure 3: Jaccard index (0=low to 1=high) evaluation of Otsu and K-means algorithms for our 30 MWT images.

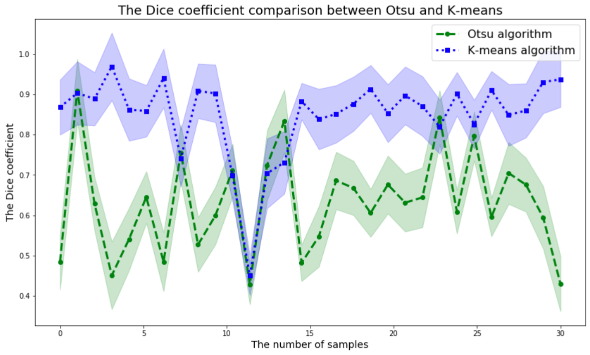


Figure 4: Dice coefficient (0=low to 1=high) evaluation of Otsu and K-means algorithms for our 30 MWT images.

derived after considerable and objective test cases. The pipeline diagram shown in Figure 6 illustrates its structure and components; the pipeline stages 2, 3, and 4 are explained next.

3.2 Image augmentation (Stage 2)

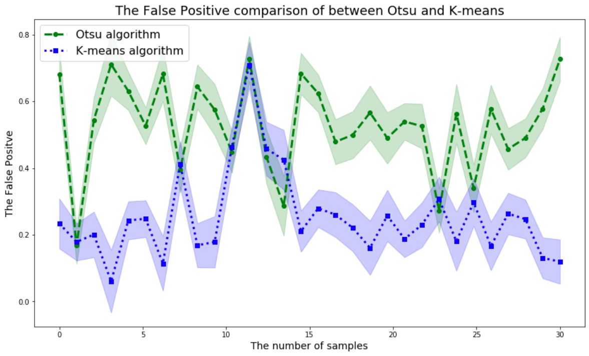


Figure 5: False positive (0=high to 1=low) evaluation of Otsu and K-means algorithms for our 30 MWT images.

The purpose of this step is to enhance the contrast and detect the boundaries displayed in our MWT images for the foams we use. Image sharpening and removing blur as well as reducing noise are common image processing tasks [19]. After repeated trials, we find that simply using the contrast adjustment function with the same level factor for all images greatly helped segmentation with k-means later. The edges become sharper and with much less noise. Basically, this operation highlights the boundaries of different color pixels which represent different levels of moisture pertaining to foams. The contrast enhancement can be easily observed when used in grayscale conversion.

3.3 Grayscale conversion (Stage 3)

Grayscale images are distinct from purely black and white images in that they can represent several shades of gray in between. They are frequently used where each pixel is a single sample indicating intensity information only. A great number of segmentation methods have been successfully implemented in grayscale rather than ordinary RGB images in tomographic systems [1]. The segmentation contributes the desirable consequences through grayscale tomographic

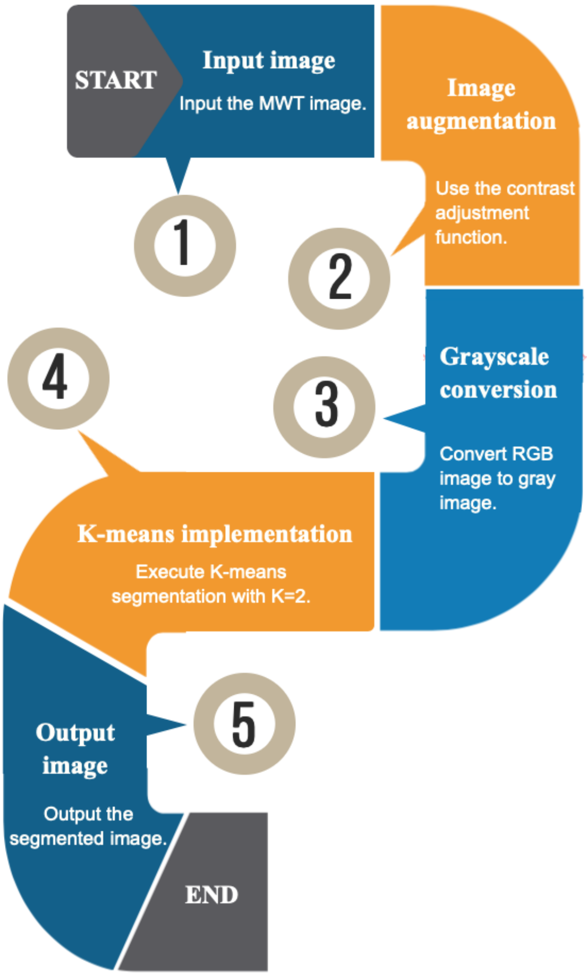

START
Input image
Input the MWT image.
1
Image augmentation
Use the contrast adjustment function.
2
Grayscale conversion
Convert RGB image to gray image.
3
4
K-means implementation
Execute K-means segmentation with K=2.
5
Output image
Output the segmented image.
END

Figure 6: The pipeline of proposed MWT Segmentation based on *K*-means (MWTS-KM) algorithm. (Step 2 and step 3 are explained in Figure 8)

images while segmenting for porous materials [19].We use the Python Imaging Library (PIL) to execute the grayscale conversion. The comparison between the input MWT and grayscale images (after contrast adjustment/augmentation), and between just contrast adjustment/augmentation and grayscale applied afterwards are displayed in Figure 8.

3.4 *K*-means implementation (Stage 4)

A *K*-means algorithm is used as the final step of our proposed algorithm to execute the segmentation stage. A fuller introduction in the following section describes how the conventional *K*-means algorithm is used in this paper for the purpose of comparison.

4 SETUP AND RESULTS

Microwave tomography mainly consists of two steps: collecting scattered electric field data around an object, and applying inversion algorithms on the scattered field data to get information of the object properties like dielectric distribution. As for the setup, the object under testing is surrounded by an array of antennas acting as transceivers. The scattered field data is collected so that each antenna is separate in the source mode, and it is assumed that it cannot receive data while acting as a source. A schematic picture proposed for the MWT is shown in Figure 7, where the antenna array is modelled as 2D line sources and a perfectly matched layer is placed over the truncated free-space.

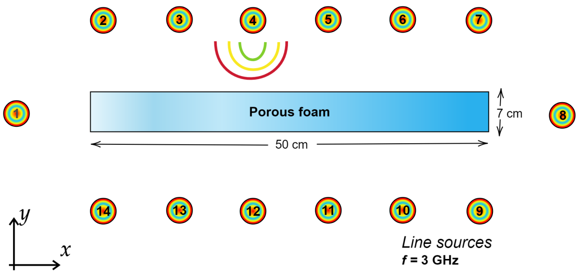


Figure 7: Model used for benchmarking purposes. The light blue box is the porous foam; green circles represent the antenna modelled as a 2D line source.

Here, the object is a piece of porous foams. Detailing the reconstruction method for estimating the moisture distribution is beyond the scope of this paper. Instead, we study automatic segmentation methodologies for the purpose of visualizing the low moisture area displayed in the MWT reconstructed images. In this study, we gather a total of 30 MWT images reconstructed from the data obtained from the sensors characterizing 30 distinct processes. The reconstructed images are all in RGB mode, and their size are kept constant throughout augmentation, grayscale conversion and three distinct segmentation phases.

4.1 Results

We conduct segmentation in all the 30 images gained from different microwave drying processes as aforementioned. In our analysis, the segmentation task is a binary classification to distinguish the foreground and the background in images. Accordingly, we intend to categorize our MWT image into two parts; low moisture area and other area. The whole segmentation period for 30 images lasts around three hours. We run all the python scripts in the full-fledged IDE–Jupyter Notebook. As aforementioned, we randomly choose 4 of those 30 samples to be shown in this paper (Figures 8 and 9) for readability and understanding due to the given limited space. The blue areas in the MWT images represent low moisture areas. Figure 8 shows a part of the pipeline of MWTS-KM, comparing the input MWT images, augmented MWT images and grayscale MWT images in which it is easy to differentiate low moisture

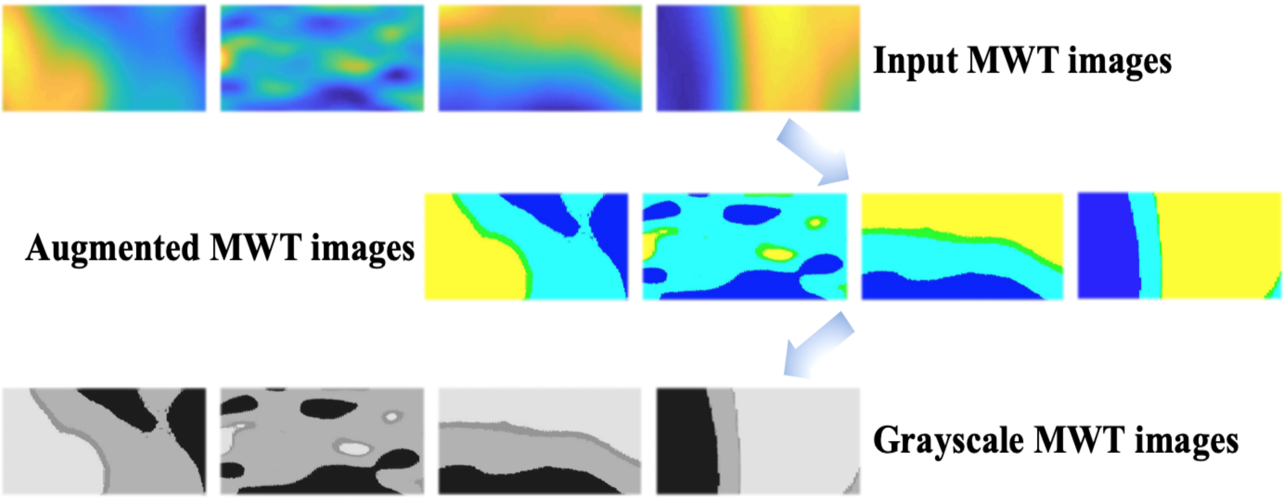


Figure 8: The comparison among input MWT images, augmented images and grayscale images. The first row stands for input MWT image, second row for augmented images and third row for grayscale images.

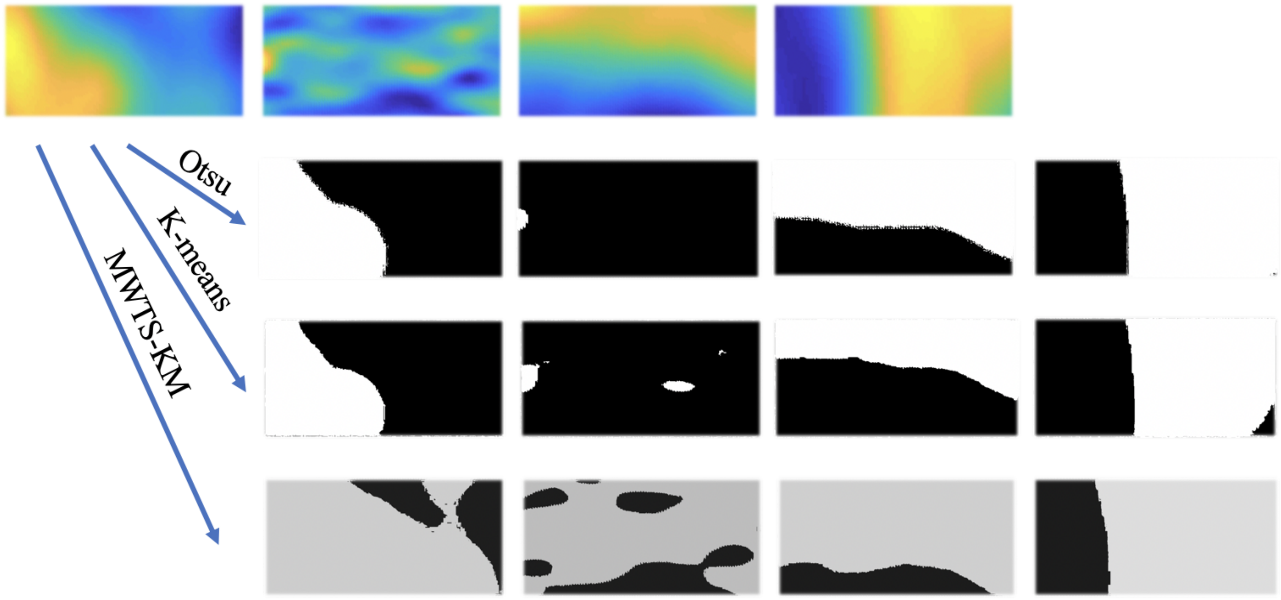


Figure 9: The comparison among three segmentation results. The first row shows the input images. The second row results from applying the Otsu algorithm on the input images. The third row results from using conventional K-means Algorithm while the last row shows output images from the MWTS-KM algorithm.

areas and other areas. The exhaustive segmentation results are shown in Figure 8 with the respect of MWTS-KM, Otsu and K-means algorithms. Dark areas in these images denote the low moisture areas that are the desirable consequences we want.

4.2 Evaluation

As Figure 9 depicts, the Otsu and K-means algorithms are not capable to visualize the low moisture areas after segmentation while MWTS-KM precisely visualized those designated areas. More specifically, there is almost no evident difference observed between the segmentation results from Otsu and K-means algorithms, both of which are ineffective in obtaining desirable consequences. However, compared to ground-truth images which are input MWT images, our proposed MWTS-KM achieves excellent accuracy in visualizing low moisture areas in terms of perception. Therefore, we provide another more convincing quantitative evaluation.

The complete quantitative evaluation output for the 30 samples is shown in Table 1, presenting the Jaccard index, Dice coefficient and false positive across all three algorithms. In terms of performance of each algorithm, MWTS-KM exceeds Otsu and K-means remarkably in each metric. While K-means performs better than Otsu in most cases, it is outperformed by MWTS-KM. In terms of the Jaccard index, Otsu produces values fluc-

Table 1: Jaccard index, Dice coefficient, false positive: Comprehensive performance comparison between 3 segmentation algorithms for 30 samples.

	Jaccard index			Dice coefficient			False positive		
	MWTS-KM	Otsu	*K*-means	**MWTS-KM**	Otsu	*K*-means	**MWTS-KM**	Otsu	*K*-means
①	**0.997**	0.319	0.766	**0.999**	0.484	0.868	**0.003**	0.682	0.234
②	**0.998**	0.834	0.823	**0.999**	0.909	0.903	**0.002**	0.168	0.178
③	**0.995**	0.458	0.799	**0.997**	0.628	0.888	**0.005**	0.544	0.201
④	**0.995**	0.291	0.939	**0.998**	0.450	0.969	**0.005**	0.711	0.061
⑤	**0.997**	0.370	0.757	**0.998**	0.540	0.862	**0.003**	0.631	0.243
⑥	**0.995**	0.476	0.752	**0.998**	0.645	0.858	**0.005**	0.527	0.248
⑦	**0.994**	0.320	0.887	**0.997**	0.484	0.940	**0.006**	0.682	0.114
⑧	**0.998**	0.606	0.589	**0.999**	0.754	0.741	**0.002**	0.395	0.412
⑨	**0.995**	0.358	0.832	**0.998**	0.527	0.908	**0.005**	0.644	0.168
⑩	**0.995**	0.427	0.821	**0.998**	0.599	0.902	**0.005**	0.575	0.179
⑪	**0.999**	0.552	0.536	**0.999**	0.712	0.698	**0.001**	0.448	0.464
⑫	**0.999**	0.272	0.291	**0.999**	0.428	0.451	**0.001**	0.728	0.709
⑬	**0.999**	0.567	0.543	**0.999**	0.724	0.704	**0.001**	0.433	0.458
⑭	**0.999**	0.714	0.576	**0.999**	0.833	0.731	**0.001**	0.286	0.425
⑮	**0.999**	0.317	0.790	**0.999**	0.482	0.882	**0.001**	0.683	0.211
⑯	**0.999**	0.377	0.721	**0.999**	0.547	0.838	**0.001**	0.624	0.279
⑰	**0.999**	0.522	0.740	**0.999**	0.686	0.851	**0.001**	0.479	0.260
⑱	**0.999**	0.501	0.779	**0.998**	0.667	0.876	**0.002**	0.500	0.221
⑲	**0.999**	0.434	0.840	**0.998**	0.605	0.913	**0.001**	0.566	0.161
⑳	**0.999**	0.510	0.743	**0.998**	0.676	0.853	**0.002**	0.490	0.257
㉑	**0.999**	0.461	0.813	**0.999**	0.631	0.897	**0.001**	0.540	0.187
㉒	**0.999**	0.475	0.771	**0.997**	0.644	0.870	**0.001**	0.526	0.230
㉓	**0.999**	0.728	0.694	**0.998**	0.843	0.819	**0.001**	0.272	0.307
㉔	**0.999**	0.437	0.819	**0.999**	0.608	0.901	**0.001**	0.563	0.181
㉕	**0.999**	0.661	0.703	**0.999**	0.796	0.825	**0.001**	0.339	0.297
㉖	**0.998**	0.424	0.834	**0.999**	0.595	0.909	**0.001**	0.577	0.166
㉗	**0.997**	0.544	0.737	**0.998**	0.704	0.848	**0.001**	0.457	0.263
㉘	**0.999**	0.510	0.754	**0.998**	0.675	0.860	**0.001**	0.490	0.247
㉙	**0.999**	0.423	0.869	**0.997**	0.594	0.930	**0.002**	0.578	0.131
㉚	**0.998**	0.274	0.881	**0.995**	0.430	0.937	**0.001**	0.727	0.119

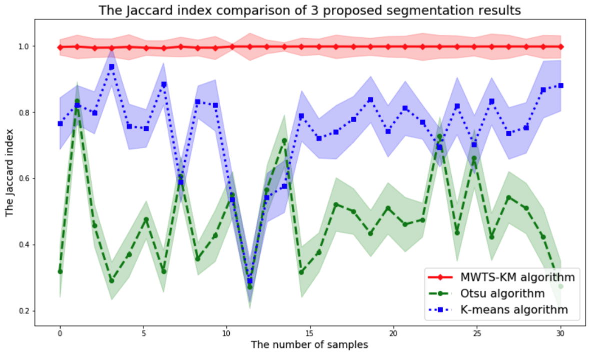


Figure 10: Jaccard index accuracy (0=low to 1=high) evaluation of MWTS-KM, Otsu, *K*-means segmentation algorithms for tested 30 MWT images.

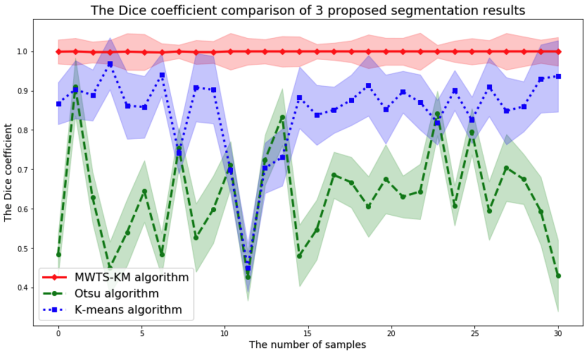


Figure 11: Dice coefficient accuracy (0=low to 1=high) evaluation of MWTS-KM, Otsu, *K*-means segmentation algorithms for test 30 images.

tuating around 0.50 for 30 samples while *K*-means produces a cluster of values around 0.80. Similarly, Otsu obtains these Dice coefficients oscillating close to 0.60 but *K*-means has a series of Dice values in proximity of about 0.82. When considering false positive, Otsu always has a higher value compared to *K*-means, which

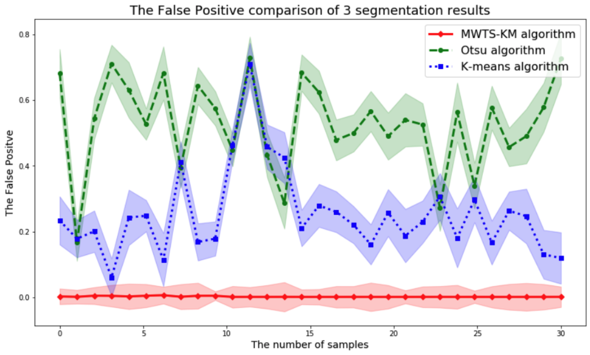


Figure 12: False positive (0=high to 1=low) evaluation of MWTS-KM, Otsu, *K*-means segmentation algorithms for test 30 images.

also confirms the above results. The visual and straightforward comparisons are displayed in Figures 10, 11, and 12.

MWTS-KM in particular outperforms Otsu and *K*-means exceedingly according to our sensitivity analysis. In fact, Otsu even has superior performance compared to *K*-means in a few experimental cases in line with quantitative evaluation. Overall, these two well-established approaches are disadvantageous in our context. It is worthwhile to observe that MWTS-KM achieves nearly one of both Jaccard index and Dice coefficient in the whole 30 segmentation results. Furthermore, the false positive values for MWTS-KM attain almost zero in each trial, which would be pertained to exceptional performance.

5 DISCUSSIONS

To summarize, we implement image segmentation technology in 30 MWT images obtained from the specialized microwave drying processes. The proposed MWTS-KM algorithm is applied to segment images to visualize the areas indicating low moisture level for porous foams in relevant context. We compare MWTS-KM with another two commonly used methods and appraise them thoroughly. *K*-means algorithm surpasses Otsu in general cases even though there is only a moderate difference between them. As shown in Figures 9, 10 and 11, there exists some intersections among Otsu and *K*-means in their performance line-plots. For example, in samples 12, 13 and 23, the Otsu algorithm is superior to *K*-means algorithm whereas it is disadvantageous compared to *K*-means algorithm in other samples. However, the red lines representing MWTS-KM indicate that our proposed algorithm has the highest Jaccard indexes and Dice coefficients with the lowest false positive values as a tremendously acceptable result.

Overall, our proposed MWTS-KM algorithm performs better than Otsu and *K*-means algorithms in both perceived and quantitative evaluation in MWT image segmentation, especially outperforming them dominantly in each case we experimented.

6 CONCLUSIONS

In this paper, firstly we prove the mature segmentation methods which have been widely used in tomographic systems are considerably suitable for MWT image segmentation. In our study, this technique is an intuitive and innovative method for visualizing the low moisture areas of foams. We have developed an entirely automatic methodology named MWTS-KM to conduct the MWT image segmentation, validating its high efficiency and high accuracy after practical experiments. This method is able to meticulously visualize the low moisture areas for foams in MWT images. Furthermore, its performance is superior to two other preeminent methods.

We acknowledge that there still are some inefficiencies in our algorithm, such as inadequate samples and lack of more complete pre-processing steps. Therefore, future work will mainly concentrate on adding more preprocessing techniques as well as testing more sample trials to enhance the robustness of the algorithm. For instance, thresholding could be a promising add-on. Additionally, we will incorporate human perception into the study, such as by changing the color schemes to compare the task performance.

7 ACKNOWLEDGMENTS

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No. 764902.

8 REFERENCES

[1] O. M. Dorgham, "Automatic body segmentation from computed tomography image," in *2017 International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)*, pp. 1–5, May 2017.

[2] N. Dhanachandra, K. Manglem, and Y. J. Chanu, "Image segmentation using k-means clustering algorithm and subtractive clustering algorithm," *Procedia Computer Science*, vol. 54, pp. 764–771, 2015.

[3] B. H. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer, K. Farahani, J. Kirby, Y. Burren, N. Porz, J. Slotboom, R. Wiest, *et al.*, "The multimodal brain tumor image segmentation benchmark (brats)," *IEEE transactions on medical imaging*, vol. 34, no. 10, pp. 1993–2024, 2014.

[4] A. Jose, S.Ravi, and M.Sambath, "Brain tumor segmentation using k-means clustering and fuzzy c-means algorithms and its area calculation,"

International Journal of Innovative Research in Computer and Communication Engineering, vol. 2, pp. 1–9, Mar 2014.

[5] Q. Mahmood, S. Li, A. Fhager, S. Candefjord, A. Chodorowski, A. Mehnert, and M. Persson, “A comparative study of automated segmentation methods for use in a microwave tomography system for imaging intracerebral hemorrhage in stroke patients,” *Journal of Electromagnetic Analysis and Applications*, vol. 7, no. 05, p. 152, 2015.

[6] K. Men, X. Chen, Y. Zhang, T. Zhang, J. Dai, J. Yi, and Y. Li, “Deep deconvolutional neural network for target segmentation of nasopharyngeal cancer in planning computed tomography images,” *Frontiers in oncology*, vol. 7, p. 315, 2017.

[7] C. Wang, W. Wang, S. Shin, and S. I. Jeon, “Comparative study of microwave tomography segmentation techniques based on gmm and knn in breast cancer detection,” in *Proceedings of the 2014 Conference on Research in Adaptive and Convergent Systems*, pp. 303–308, ACM, 2014.

[8] P. Mojabi, M. Ostadrahimi, L. Shafai, and J. LoVetri, “Microwave tomography techniques and algorithms: A review,” in *2012 15 International Symposium on Antenna Technology and Applied Electromagnetics*, pp. 1–4, June 2012.

[9] Z. Wu and H. Wang, “Microwave tomography for industrial process imaging: Example applications and experimental results.,” *IEEE Antennas and Propagation Magazine*, vol. 59, pp. 61–71, Oct 2017.

[10] A. Fhager, M. Gustafsson, and S. Nordebo, “Image reconstruction in microwave tomography using a dielectric debye model,” *IEEE Transactions on Biomedical Engineering*, vol. 59, pp. 156–166, Jan 2012.

[11] M. Pastorino, *Microwave imaging*, vol. 208. John Wiley & Sons, 2010.

[12] A. H. Sihvola, *Electromagnetic mixing formulas and applications*. No. 47, Iet, 1999.

[13] Y. Zhang, Y. Ma, A. Omrani, R. Yadav, M. Fjeld, and M. Fratarcangeli, “Automatic image segmentation for microwave tomography (mwt) from implementation to comparative evaluation,” in *Proceedings of the 12th International Symposium on Visual Information Communication and Interaction*, pp. 1–2, 2019.

[14] S. Neeraj and L. M. Aggarwal, “Automated medical image segmentation techniques,” *J Med Phys*, vol. 35, pp. 3–14, Jan-Mar 2010.

[15] M. Shoaib, R. Naseem, and A. H. Dar, “Automated segmentation of lungs incomputed tomographic images,” *European Journal of Scientific Research*, vol. 98, pp. 45–54, Mar 2013.

[16] N. Otsu, “A threshold selection method from gray-level histograms,” *IEEE transactions on systems, man, and cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[17] H. Permuter, J. Francos, and I. Jermyn, “A study of gaussian mixture models of color and texture features for image classification and segmentation,” *Pattern Recognition*, vol. 39, no. 4, pp. 695 – 706, 2006. Graph-based Representations.

[18] S. Wu, S. Yu, L. Zhuang, X. Wei, M. Sak, N. Duric, J. Hu, and Y. Xie, “Automatic segmentation of ultrasound tomography image,” *BioMed research international*, vol. 2017, 2017.

[19] A. P. Sheppard, R. M. Sok, and H. Averdunk, “Techniques for image enhancement and segmentation of tomographic images of porous materials,” *Physica A: Statistical Mechanics and its Applications*, vol. 339, no. 1, pp. 145 – 151, 2004. Proceedings of the International Conference New Materials and Complexity.

[20] M. Kass, A. Witkin, and D. Terzopoulos, “Snakes: Active contour models,” *International Journal of Computer Vision*, vol. 1, pp. 321–331, Jan 1988.

[21] L. Vincent and P. Soille, “Watersheds in digital spaces: An efficient algorithm based on immersion simulations,” *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 13.

[22] A. Rashno, D. D. Koozekanani, P. M. Drayna, B. Nazari, S. Sadri, H. Rabbani, and K. K. Parhi, “Fully automated segmentation of fluid/cyst regions in optical coherence tomography images with diabetic macular edema using neutrosophic sets and graph algorithms,” *IEEE Transactions on Biomedical Engineering*, vol. 65, no. 5, pp. 989–1001, 2018.

[23] F. G. Venhuizen, B. van Ginneken, B. Liefers, M. J. van Grinsven, S. Fauser, C. Hoyng, T. Theelen, and C. I. Sánchez, “Robust total retina thickness segmentation in optical coherence tomography images using convolutional neural networks,” *Biomed. Opt. Express*, vol. 8, pp. 3292–3316, Jul 2017.

[24] E. J. Joseph, K. A. H. Ping, K. Kipli, D. A. A. Mat, S. Sahrani, D. N. A. Zaidel, M. I. Sariphn, and M. H. Marhaban, “Integration of image segmentation method in inverse scattering for brain tumour detection,” *Progress In Electromagnetics Research*, vol. 61, pp. 111–122, 2017.

[25] H. J. Vala and A. Baxi, “A review on otsu image segmentation algorithm,” *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, vol. 2, no. 2, pp. 387–389, 2013.

[26] J. Zhang and J. Hu, "Image segmentation based on 2d otsu method with histogram analysis," in *2008 International Conference on Computer Science and Software Engineering*, vol. 6, pp. 105–108, IEEE, 2008.

[27] Y. Wang, "Improved otsu and adaptive genetic algorithm for infrared image segmentation," in *2018 Chinese Control And Decision Conference (CCDC)*, pp. 5644–5648, IEEE, 2018.

[28] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[29] S. Ray and R. H. Turi, "Determination of number of clusters in k-means clustering and application in colour image segmentation," in *Proceedings of the 4th international conference on advances in pattern recognition and digital techniques*, pp. 137–143, Calcutta, India, 1999.

[30] N. Sharma and L. M. Aggarwal, "Automated medical image segmentation techniques," *Journal of medical physics/Association of Medical Physicists of India*, vol. 35, no. 1, p. 3, 2010.

ISSN 2464-4617 (print)
ISSN 2464-4625 (DVD)

Computer Science Research Notes
CSRN 3001

WSCG2020 Proceedings

Fast View Synthesis for Immersive Video Systems

Jakub Stankowski
Institute of Multimedia Telecommunications
Poznań University of Technology
Polanka 3
61-131 Poznań, Poland
jakub.stankowski@put.poznan.pl

Adrian Dziembowski
Institute of Multimedia Telecommunications
Poznań University of Technology
Polanka 3
61-131 Poznań, Poland
adrian.dziembowski@put.poznan.pl

ABSTRACT

Immersive video has become a popular research topic recently. However, there are no fast immersive video processing methods, which could be used in practical immersive video systems. In this paper the real-time CPU-based virtual view synthesis method is presented. The proposed method allows a viewer to freely navigate within acquired scene without necessity of using dedicated FPGA devices or powerful graphic cards. Presented view synthesis method can be used in practical immersive video systems, even for ultra-high resolution sequences. In order to present usefulness of proposed method, several implementations and use cases are discussed in the paper.

Keywords
Virtual view synthesis, immersive video systems, real-time video processing.

1. INTRODUCTION

In this paper we deal with the virtual view synthesis for immersive video systems. Such kind of systems allow a viewer to immerse into a scene, i.e. to virtually navigate within a scene that was captured by a set of arbitrarily located cameras [Goo12][Sta18][Zit04] (Fig. 1).

In order to provide the possibility of smooth navigation of a user, his or her viewpoint cannot be limited only to images captured by multiple cameras – a user should be able to watch the scene from any, arbitrarily chosen position (orange camera in Fig. 1). In order to generate additional images, the virtual view synthesis operation should be used [Sun10].

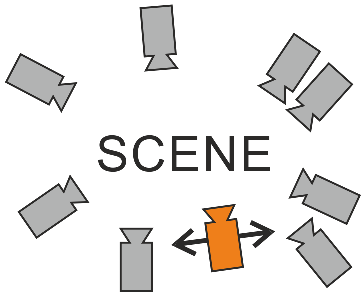


Figure 1. Idea of the immersive video system; gray – real cameras, orange – virtual camera.

There are numerous virtual view synthesis methods and algorithms described in literature (e.g. [Dzi19][Fac18][Nia18][Sen18][Wan19]). However, they cannot be used in the practical immersive video system because of the processing time. When a user of the immersive video system demands a particular virtual view, the view has to be generated immediately in order to eliminate delays between user's action and viewpoint change. Therefore, all the processing has to be performed in the real time.

The real-time virtual view synthesis methods are also known, but they usually require dedicated FPGA [Aki15][Li19][Wan12] or VLSI [Hua19] devices or powerful graphic cards [Non18][Yao16][Zha17].

In the practical, consumer immersive video system, developed for the entire spectrum of final users, it may disqualify users, as they do not have appropriate hardware due to cost or compatibility.

In this paper, the real-time virtual view synthesis for CPU is presented. So far, only one this kind of method was described in literature [Dzi18]. It was able to process FullHD sequences in real-time, but only for reduced output resolution (i.e. qHD), not to mention higher resolutions (4K). The method presented in this paper allows synthesis of UltraHD content in the real-time, what makes it usable also for the most recent immersive video systems.

2. VIEW SYNTHESIS ALGORITHM

The practical view synthesis method should meet two main requirements. At first, it has to be fast enough to be used in the real-time, consumer immersive video system. Secondly, the quality of synthesized virtual views should be as high as possible.

https://doi.org/10.24132/CSRN.2020.3001.16

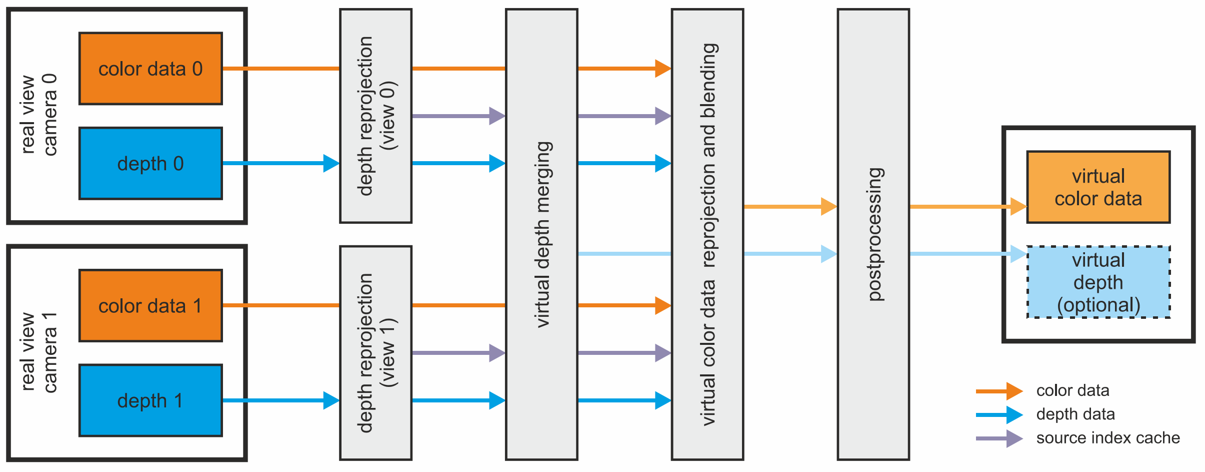


Figure 2. Block diagram presenting data flow in proposed algorithm.

In order to obtain good quality of virtual views, we decided to develop a backward-type synthesis [Duh13][Shi13]. The major advantage of such synthesis type is to admit filtering of the reprojected depth map before texture reprojection. However, typical backward-type synthesis requires two steps of reprojection: depth from input view to the virtual view and texture in the opposite direction, which makes it slower. In the proposed approach, the backward-type synthesis was modified in order to reduce the number of reprojections to one.

The proposed view synthesis algorithm consists of four main stages (Fig. 2): depth reprojection (performed separately for both real views), depth merging, color data reprojection and preprocessing. The purpose of two first operations is to create a depth map of the virtual view. Then, this depth map is used for reprojecting color data to the virtual view. Finally, the virtual view is postprocessed in order to achieve the highest quality. All the steps of proposed algorithm are described in following subsections.

Depth reprojection

In the first stage only input depth maps are analyzed. Reprojection of each pixel of the input view i is conducted by multiplication of a vector containing pixel's position (x_i, y_i) and depth (z_i) and a homography matrix $\mathbf{H}_{i,v}$:

$$\begin{bmatrix} x_v \\ y_v \\ z_v \\ 1 \end{bmatrix} = \mathbf{H}_{i,v} \cdot \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix}.$$

The homography matrix is a 4×4 matrix defined as multiplication of projection matrix of virtual view v and inverted projection matrix of input view i [Hey08].

This operation requires 16 multiplications and 12 additions for each processed pixel. In the proposed algorithm it was optimized, resulting in 4 multiplications and 3 additions per pixel, with additional 4 multiplications and 3 additions for each column and each row. However, calculations for columns and rows are performed once in a preprocessing step and their results are then stored in look up tables (LUTs).

Depth merging

In the first stage, both input views are processed separately. It results in two virtual depth maps, each containing depth values reprojected from single input depth map.

In the second stage, both virtual depth maps are merged. Within the merging operation, three cases are possible for each pixel.

If the pixel was not reprojected from any input view – it will become empty in the merged depth map. If it was reprojected from only one view – the merged depth map will contain value reprojected from that view. Finally, if there are two depth candidates for one pixel, smaller depth (closer to a camera, because it occludes further one) will be copied into the merged view.

Color data reprojection and blending

In the third stage, the actual virtual view is created. Each pixel of the virtual view is calculated by analyzing values in corresponding pixels in input views. In the typical backward-type synthesis, it would require an additional reprojection step. In the proposed approach, positions of corresponding pixels within input views are stored in source index cache, which reduces this operation to memory reading.

If the pixel was visible in only one input view, its color is copied from that input view. If it was reprojected from both input views, its color in the virtual view is calculated as an average of colors in both input views. If it was not visible in any input view, it remains a hole.

Postprocessing

The last stage of the proposed algorithm allows enhancement of the quality of the synthesized virtual view. It consists of two main steps: filtering and inpainting.

In the first step, the virtual view and corresponding depth map are filtered. They are filtered at the same time, but only depth values are being analyzed in this step. The virtual view is filtered in order to remove small artifacts such as object discontinuities or single pixels with wrong depth caused by blurred edges in input depth maps.

In order to perform fast filtration, it is checked for each pixel, whether it is surrounded by pixels with different depth. In horizontal filtering step, it is checked if the depth of analyzed pixel is much higher or much lower, than depth of its left and right. If so, color and depth of analyzed pixel is replaced by color and depth of its right. The vertical filtering step is performed based on top and bottom neighbors and filtered pixel is eventually replaced by bottom neighbor.

In the second step inpainting of the virtual view is performed. This step is crucial for the virtual view synthesis [Ber00][Cri04][Dar10] because it allows holes (i.e. areas without information reprojected from input views) filling in the virtual view. In the proposed approach the fast depth-based 4-way inpainting method is used. For each empty pixel four closest pixels in 4 directions (left, right, top and bottom neighbors) are compared. Color of the pixel with closest depth is copied to the analyzed one.

3. IMPLEMENTATION AND OPTIMIZATION DETAILS

The proposed algorithm has been implemented in portable C++ language, therefore (excluding vectorized version mentioned further) can be ported to almost any hardware platform. The single threaded implementation is based on the one described in [Dzi18]. Nonetheless, significant improvements have been developed. The proposed implementation allows processing high bit-depth sequences (up to 14 bits per pixel) and high precision depth maps (up to 16 bits per depth element).

The algorithmic optimization includes following techniques:

1. memory access optimization by reducing redundant loads/stores and using prefetch friendly data layouts,
2. usage of local buffers and pre-calculated LUTs,
3. reduction of the number of required multiplication and additional operations in depth reprojection stage.

Moreover, the implementation eliminates projection related computations from virtual view projection stage and reduce its computational complexity. During preceding (depth reprojection) stage, the source location of depth element is cached and reused in view reprojection.

Implementation using vector extensions

The majority of modern processors include some sort of vector processing units, allowing computations on several values at once. The usage of vector instructions allows a significant reduction of computation times, especially for execution bound algorithms.

In case of virtual view synthesis, depth reprojection is the most computationally complex. Therefore we decided to develop vectorized implementation of depth reprojection stage. We concentrated on AVX2 and AVX512 extensions available in modern x86-64 processors. The AVX2 and AVX512 extensions [Dem13] enable operating on 256 bit (containing 8 single precision floats) and 512 bit (containing 16 single precision floats) vector respectively. Moreover, both extensions allow to use FMA instructions [Qui07] which are very useful in the reprojection stage.

The AVX512 vector is twice as wide as AVX2 one, allowing twice as much data at one clock cycle. In addition, AVX512 instruction set admits mask registers and per-lane predication, both to write more efficient code and to reduce register pressure [Dem13].

Parallel implementation

Another approach to speed up the virtual view synthesis is to parallelize computation by using multi-threaded implementation.

Most of the synthesis-related operations, like depth merging, color data reprojection and merging and postprocessing, could be easily parallelized by dividing the picture into arbitrary number of slices and processing each slice by dedicated thread.

Unfortunately, the most complex operation in the proposed algorithm, namely the depth reprojection, is not easy to parallelize. The reason is the risk of data race caused by unpredictable location of a reprojected depth element. Therefore, there is no possibility to simply compute each of input depth slices by separate thread.

The simplest approach, presented in [Dzi18] is to perform reprojection of each depth in a separate thread i.e. the first thread processes the “depth 0”, while the second processes the “depth 1”, and so on. Unfortunately, this approach allows parallelization by factor of 2 only and is insufficient in case of modern multicore processors.

Independent Projection Targets

In order to improve the parallelization factor for depth reprojection, the Independent Projection Targets (IPT) approach has been proposed. The idea of IPT is to use separate buffers (projection targets) for each of processing threads (Figure 3). Both reprojected depth and source index cache are buffered. The usage of IPT removes the restriction for depth reprojection parallelization level and allows using all available processing threads.

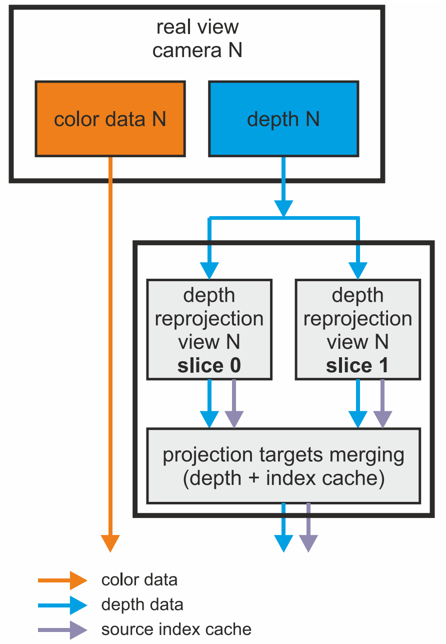

real view
camera N
color data N
depth N
depth reprojection view N slice 0
depth reprojection view N slice 1
projection targets merging (depth + index cache)
color data
depth data
source index cache

Figure 3. Independent Projection Targets.

The drawback of IPT is the necessity of additional operations to merge results from all projection targets, as well as the increase of memory footprint due to excessive buffering. Nonetheless, the additional complexity of depth merging stage does not offset the reduced complexity of depth reprojection stage.

4. METHODOLOGY

Test sequences

The test set contained three miscellaneous high-resolution test sequences (Fig. 4):

1. PoznanFencing, FullHD resolution, sparse arc arrangement [Dom16],
2. TechnicolorPainter, 2K resolution, dense linear arrangement [Doy17],
3. PoznanBasketball, FullHD resolution, sparse linear arrangement [Dom18].

Two of them (1 and 2) are commonly used in the research and developing immersive video standards. The third one was placed into the set because of very different content/characteristics – it contains a fragment of basketball match, what could be one of possible use cases of immersive video systems.

In order to simulate virtual view synthesis for UltraHD (4K) input views, one of the experiments required UltraHD sequence. Because of lack of such test material, resolution of TechnicolorPainter sequence was increased. Remaining samples of input view were calculated using 1st order interpolation, while samples of depth maps – using 0th order interpolation (in order to avoid introducing non-existent depth values at the objects' edges – if linear interpolation will be used, physical edges of the objects will be destroyed, e.g. between a pixel representing a person and a pixel representing a wall behind, there would be a pixel with averaged depth, representing physically non-existing object).

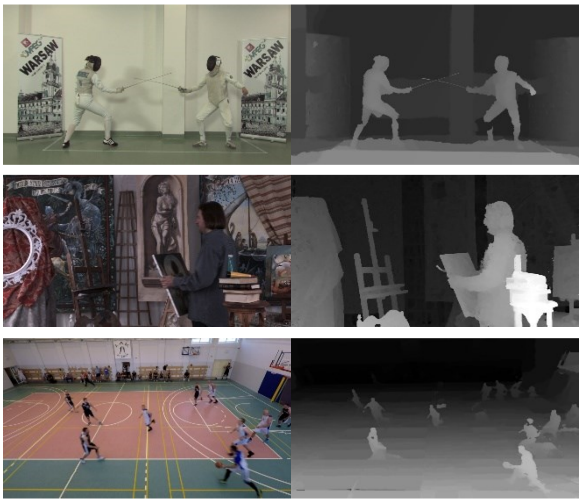


Figure 4. Input views and corresponding depth maps for (from top): PoznanFencing, TechnicolorPainter, PoznanBasketball.

Evaluated implementations

Experiments were performed on 10 implementations. Implementations were divided into 4 types: R – the reference implementation, which does not include any optimizations and is treated as a base for comparison with others; A – the optimized implementation; B – optimized and vectorized implementation using AVX2 instruction set; C – optimized and vectorized implementation using AVX512 instruction set.

Moreover, each implementation (except for reference one) was tested in 3 versions: single-threaded (1), multi-threaded (2) and multi-threaded with IPT (3).

Quality evaluation

In order to evaluate the quality of virtual views synthesized using presented algorithm, 5 objective quality metrics were used: PSNR, Multi-Scale SSIM (MS-SSIM) [Wan03], Visual Information Fidelity (VIF) [She06], Video Multimethod Assessment Fusion (VMAF) [Li16] and IVPSNR, which is ISO/IEC MPEG's metric for immersive video [MPEG19].

Implementation	Implementation features				Processing time [ms]				
	Optimized	Vectorized	Multi-threaded	Independent Projection Targets	Depth projection	Depth merging	View projection	Post-processing	Total
VSRS (state-of-the-art view synthesis method)					–	–	–	–	**2581.12**
R	–	–	–	–	127.41	0.83	14.19	18.84	**161.27**
A1	✓	–	–	–	39.25	0.82	15.81	19.30	**75.18**
A2	✓	–	✓		35.30	0.32	4.29	6.04	**45.95**
A3	✓	–	✓	✓	23.42	2.53	4.11	5.63	**35.69**
B1	✓	AVX2	–	–	15.77	0.79	10.98	18.73	**46.26**
B2	✓	AVX2	✓	–	18.62	0.32	2.62	5.47	**27.03**
B3	✓	AVX2	✓	✓	10.16	1.85	2.59	5.66	**20.26**
C1	✓	AVX512	–	–	10.26	0.80	11.04	19.03	**41.13**
C2	✓	AVX512	✓	–	12.83	0.32	2.62	5.58	**21.35**
C3	✓	AVX512	✓	✓	7.62	1.74	2.59	5.41	**17.35**

Table 1. Comparison of all implementations (TechnicolorPainter sequence, FullHD → FullHD scenario)

Test sequence			Implementation	Processing time [ms]				
Sequence name	Input / output resolution	Camera arrangement		Depth projection	Depth merging	View projection	Post-processing	Total
TechnicolorPainter	2048×1088	dense linear	C3	7.62	1.74	2.59	5.41	**17.35**
PoznanBasketball	1920×1080	sparse linear	C3	4.28	1.77	4.04	4.28	**14.37**
PoznanFencing2	1920×1080	sparse arc	C3	4.18	1.78	5.77	5.20	**16.94**

Table 2. Comparison of all test sequences (FullHD → FullHD scenario, C3 implementation)

Test sequence			Implementation	Processing time [ms]				
Sequence name	Input resolution	Output resolution		Depth projection	Depth merging	View projection	Post-processing	Total
TechnicolorPainter	4096×2176	4096×2176	C3	17.60	6.86	7.92	11.35	**43.74**
TechnicolorPainter	4096×2176	2048×1088	C3	21.68	1.71	2.72	5.08	**31.20**

Table 3. UltraHD input sequence (C3 implementation)

Quality metric	Sequence name / View synthesis algorithm					
	TechnicolorPainter		PoznanBasketball		PoznanFencing2	
	VSRS	Proposed	VSRS	Proposed	VSRS	Proposed
Y-PSNR	35.94 dB	36.69 dB	28.75 dB	29.27 dB	28.26 dB	28.88 dB
C_B-PSNR	46.81 dB	47.72 dB	40.13 dB	41.76 dB	44.72 dB	45.42 dB
C_R-PSNR	46.78 dB	47.04 dB	39.48 dB	37.08 dB	39.50 dB	44.76 dB
VIF	0.574	0.615	0.456	0.482	0.272	0.270
VMAF	87.48	91.24	59.53	61.75	56.77	57.20
MS-SSIM	0.981	0.984	0.949	0.955	0.936	0.933
IVPSNR	45.94 dB	47.56 dB	36.26 dB	36.60 dB	40.07 dB	40.29 dB

Table 4. Virtual view synthesis quality

The proposed method was compared to commonly used state-of-the-art method, developed by ISO/IEC MPEG group, namely View Synthesis Reference Software (VSRS) [Sen17].

Synthesis time evaluation

The computational complexity of each implementation was evaluated by measuring processing time. Moreover, detailed statistics for each synthesis stage have been gathered.

The calculations were performed on the desktop computer equipped with 10-core CPU based on the "Skylake-X" microarchitecture. Time measurements were made using precision time stamps according to [MDNL20].

5. EXPERIMENTAL RESULTS

Comparison of all described implementations has been presented in Table 1. The results are presented for TechnicolorPainter sequence (as the worst case of all considered sequences). In the case of reference implementation, the synthesis of virtual view frame takes ~160 ms which corresponds to ~6 frames per second (FPS). This is obviously insufficient for real-time purposes. The fastest implementation – C3 (optimized, multi-threaded and with AVX512 usage) requires only 17.35 ms to synthesize one frame (resulting in 57 FPS).

Usage of vectorized implementation allows reducing depth reprojection time from ~39 ms to ~16 ms and ~10 ms for AVX2 and AVX512 respectively.

The parallel processing significantly reduces computation time for view projection and post-processing stages. In the case of depth projection, parallel processing without IPT does not seem beneficial. Usage of IPT significantly speeds up the projection stage, however it increases the complexity of depth merging stage. Nevertheless, the IPT reduces total synthesis time.

The computation time for state-of-the-art technique (VSRS) oscillate near 2.5 seconds which makes the proposed technique two orders of magnitude faster when compared to VSRS.

Table 2 includes results for all test sequences. It is noticeable that proposed synthesizer retains its performance regardless of input sequence type. Moreover, the synthesis time for sequences with sparse camera arrangement (especially PoznanBasketball) is even shorter than for previously analyzed TechnicolorPainter.

Additional results (Table 3) have been gathered for simulated UltraHD (4K) data and measured as ~23 FPS and ~32 FPS for UltraHD and FullHD target respectively. The synthesis with UltraHD source and FullHD target resolution could be considered as typical use for transmission to mobile devices.

Comparison with state-of-the-art reference technique (VSRS) shows similar synthesized image quality for both VSRS and the proposed technique (Table 4, Fig. 5). Therefore, no quality degradation was introduced during development of fast synthesis algorithm.

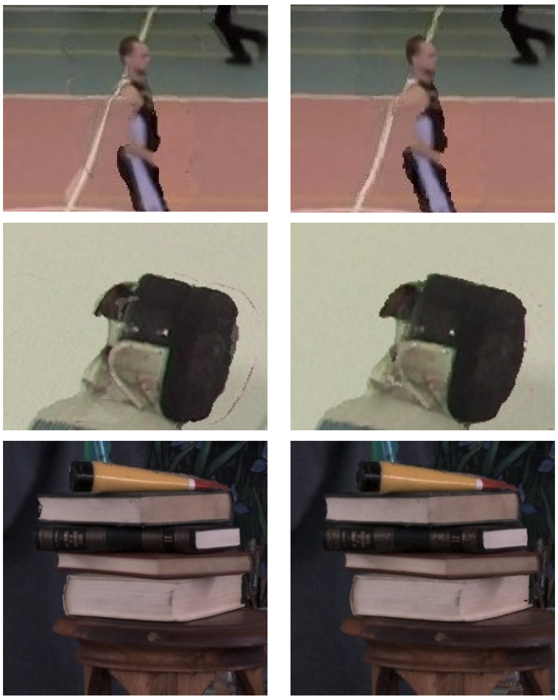


Figure 5. Fragments of virtual views synthesized using VSRS (left) and proposed method (right).

6. CONCLUSIONS

The real-time virtual view synthesis method has been presented in this paper. The experimental results show, that CPU-based implementation of the real-time view synthesis assuring good-quality virtual views is possible, even for UltraHD sequences. Therefore, it will be possible to develop cheap, consumer immersive video systems in the near future.

7. ACKNOWLEDGMENTS

This work was supported by the Ministry of Science and Higher Education.

8. REFERENCES

[Aki15] Akin, A., Capoccia, R., Narinx, J., Masur, J., Schmid, A., and Leblebici, Y. Real-time free viewpoint synthesis using three-camera disparity estimation hardware. 2015 IEEE International Symposium on Circuits and Systems (ISCAS), Lisbon, pp. 2525-2528, 2015.

[Ber00] Bertalmio, M., Sapiro, G., Caselles, V., and Ballester, C. Image inpainting. SIGGRAPH 2000, New Orlean, USA, 2000.

[Cri04] Criminisi, A., Perez, P., and Toyama, K. Region filling and object removal by exemplar-based image inpainting. IEEE Transactions on Image Processing, vol. 13, no. 9, pp. 1200-1212, 2004.

[Dar10] Daribo, I., and Pesquet-Popescu, B. Depth-aided image inpainting for novel view synthesis. 2010 IEEE International Workshop on Multimedia Signal Processing, Saint Malo, France, 2010.

[Dem13] Demikhovsky, E. Intel® AVX-512 Architecture. Comprehensive vector extension for HPC and enterprise, LLVM Developers' Meeting, San Francisco, USA, 2013.

[Dom16] Domański, M., Dziembowski, A., Grzelka, A., Łuczak, A., Mieloch, D., Stankiewicz, O., and Wegner, K. Multiview test video sequences for free navigation exploration obtained using pairs of cameras. ISO/IEC JTC1/SC29/WG11 MPEG, M41994, M38247, Geneva, Switzerland, 2016.

[Dom18] Domański, M., Dziembowski, A., Grajek, T., Grzelka, A., Klimaszewski, K., Mieloch, D., Ratajczak, R., Stankiewicz, O., Siast, J., Stankowski, J., and Wegner, K. Free-viewpoint Television demonstration for sports events. ISO/IEC JTC1/SC29/WG11 MPEG, M41994, Gwangju, Korea, 2018.

[Doy17] Doyen, D., Langlois, T., Vandame, B., Babon, F., Boisson, G., Sabater, N., Gendrot, R., and Schubert, A. Light field content from 16-camera rig. ISO/IEC JTC1/SC29/WG11 MPEG, M40010, Geneva, Switzerland, 2017.

[Duh13] Du-Hsiu, L., Hsueh-Ming, H., and Yu-Lun, L. Virtual view synthesis using backward depth warping algorithm. Picture Coding Symposium, PCS 2013, San Jose, USA, 2013.

[Dzi18] Dziembowski, A., and Stankowski, J. Real-time CPU-based virtual view synthesis. 2018 International Conference on Signals and Electronic Systems (ICSES), Kraków, Poland, 2018.

[Dzi19] Dziembowski, A., Mieloch, D., Stankiewicz, O., Domański, M., Lee, G., and Seo, J. Virtual view synthesis for 3DoF+ video. 2019 Picture Coding Symposium (PCS), Ningbo, China, 2019.

[Fac18] Fachada, S., Bonatto, D., Schenkel, A., and Lafruit, G. Depth image based view synthesis with multiple reference views for virtual reality. 3DTV-Conference: The True Vision – Capture, Transmission and Display of 3D Video (3DTV-CON), Helsinki, Finland, 2018.

[Goo12] Goorts, P., Dumont, M., Rogmans, S., and Bekaert, P. An end-to-end system for free viewpoint video for smooth camera transitions. 2012 International Conference on 3D Imaging (IC3D). Liege, Belgium, 2012.

[Hey08] Heyden, A., and Pollefeys, M. Multiple view geometry. In: Emerging Topics in Computer Vision. Prentice Hall, pp. 63-75, 2008.

[Hua19] Huang, H., Wang, Y., Chen, W., Lin, P. and Huang, C. System and VLSI implementation of phase-based view synthesis. 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, United Kingdom, pp. 1428-1432, 2019.

[Li16] Li, Z., Aaron, A., Katsavounidis, I., Moorthy, A., and Manohara, M. Toward a practical perceptual video quality metric. Netflix Technology Blog, 2016.

[Li19] Li, Y., Claesen, L., Huang, K., and Zhao, M. A real-time high-quality complete system for depth image-based rendering on FPGA. IEEE Transactions on Circuits and Systems for Video Technology, vol. 29, no. 4, pp. 1179-1193, 2019.

[MDNL20] Microsoft Developer Network Library. Acquiring high-resolution time stamps. https://msdn.microsoft.com/enus/library/windows/desktop/dn553408, 2020.

[MPEG19] Software manual of IV-PSNR for Immersive Video. ISO/IEC JTC1/SC29/WG11 MPEG, W18709, Göteborg, Sweden, 2019.

[Nia18] Nian, Z., and Jung, C. High-quality virtual view synthesis for light field cameras using multi-loss convolutional neural networks. 2018 25th IEEE International Conference on Image Processing (ICIP), Athens, Greece, 2018.

[Non18] Nonaka, K., Watanabe, R., Chen, J., Sabirin, H., and Naito, S. Fast plane-based free-viewpoint synthesis for real-time live streaming. 2018 IEEE Visual Communications and Image Processing (VCIP), Taichung, Taiwan, pp. 1-4, 2018.

[Qui07] Quinnell, E., Swartzlander, E.E., and Lemonds, C. Floating-point fused multiply-add architectures. 41 Conference on Signals, Systems and Computers, Pacific Grove, pp. 331-337, 2007.

[Sen17] Senoh, T., Yamamoto, K., Tetsutani, N., Yasuda, H., and Wegner, K. View Synthesis Reference Software (VSRS) 4.2 with improved inpainting and hole filing. ISO/IEC JTC1/SC29/WG11 MPEG, M40657, Hobart, Australia, 2017.

[Sen18] Senoh, T., Tetsutani, N., and Yasuda, H. Depth estimation and view synthesis for immersive media. 2018 International Conference on 3D Immersion (IC3D), Brussels, Belgium, 2018.

[She06] Sheikh, H.R., and Bovik, A.C. Image information and visual quality. IEEE Transactions on Image Processing, vol. 15, no. 2, pp. 430-444, 2006.

[Shi13] Shimizu, S., Sugimoto, S., Kimata, H., and Kojima, A. Backward view synthesis prediction using virtual depth map for multiview video plus depth map coding. 2013 Visual Communications and Image Processing (VCIP), Kuching, Malaysia, 2013.

[Sta18] Stankiewicz, O., Domański, M., Dziembowski, A., Grzelka, A., Mieloch, D., Samelak, and J. A Free-viewpoint Television system for horizontal virtual navigation. IEEE Transactions on Multimedia, vol. 20, no. 8, pp. 2182-2195, 2018.

[Sun10] Sun, W., Xu, L., Au, O.C., Chui, S.H., and Kwok, C.W. An overview of free viewpoint Depth-Image-Based-Rendering (DIBR). Proceedings of the Second APSIPA Annual Summit and Conference, Biopolis, Singapore, pp. 1023-1030, 2010.

[Wan03] Wang, Z., Simoncelli, E.P., and Bovik, A.C. Multiscale structural similarity for image quality assessment. The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, vol. 2, pp. 1398-1402, 2003.

[Wan12] Wang, J., and Roeningen, L.A. Real time believable stereo and virtual view synthesis engine for autostereoscopic display. 2012 International Conference on 3D Imaging (IC3D). Liege, Belgium, 2012.

[Wan19] Wang, S., and Wang, R. Robust view synthesis in wide-baseline complex geometric environments. 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, United Kingdom, 2019.

[Yao16] Yao, L., Liu, Y., and Xu, W. Real-time virtual view synthesis using light field. EURASIP Journal on Image and Video Processing, vol. 2016, pp. 1-10, 2016.

[Zit04] Zitnick, C.L., Kang, S.B., Uyttendaele, M., Winder, S., and Szeliski, R. High-quality video view interpolation using a layered representation. ACM Transactions on Graphics, vol. 3, pp. 600-608, 2004.

[Zha17] Zhang, L., Li, Y., Zhu, Q., and Li, M. Generating virtual images for multi-view video. Chinese Journal of Electronics, vol. 26, no. 4, pp. 810-813, 2017.

Estimation of the Optimum Depth Quantization Parameter for a Given Bitrate of Multiview Video plus Depth in 3D-HEVC Coding

Yasir Al-Obaidi
Institute of Multimedia Telecommunications
Poznań University of Technology
60-965, Poznań, Poland
yasir.al-obaidi@doctorate.put.poznan.pl

Tomasz Grajek
Institute of Multimedia Telecommunications
Poznań University of Technology
60-965, Poznań, Poland
tomasz.grajek@put.poznan.pl

ABSTRACT

One of the most popular data formats to represent natural immersive visual data is the MVD (Multiview Video plus Depth) format. The representation of a three-dimensional scene requires a huge amount of data in the form of a dense set of views accompanied by high-quality depth maps. All of this data need to be transmitted over the network to a viewer. Therefore, a question arises, how to allocate the bitrate between views and depth maps to obtain the maximal quality for a given bit budget. In the paper, a simple model for optimum bitrate allocation between color and depth data in 3D-HEVC coding is proposed. The provided model for quantization parameters allows better bitrate division between color and depth data, which leads to a significant (23-35%) bitrate reduction of the total bitrate of the multiview stream. At the same time, it preserves the same quality of synthesized virtual views in comparison to the common test condition (CTC) recommendation, which is considered as a well-established reference encoder configuration.

Keywords

Multiview video plus depth (MVD), Bit allocation, 3D-HEVC, 3D video coding.

1. INTRODUCTION

Immersive visual media are a top research topic nowadays, since they can provide, for example, real depth perception and realistic vision. Immersive visual data [Isg04] may refer to both computer-generated and natural content.

In immersive applications like free-viewpoint television (FTV) [Tan12, Sta18] additional views of a scene have to be generated based on the recorded ones. This allows a viewer to freely walk through and look around in the recorded scenes. Additional views are commonly generated via view synthesis techniques [Fen04], [Zin10].

One of the most popular data formats to represent natural immersive visual data is the MVD (Multiview Video plus Depth) format [Mul11]. For natural content, MVD means that there are some views acquired with synchronized cameras placed in quite arbitrary positions around a scene and corresponding depth maps. The depth maps may be acquired directly by specialized cameras [Par20, Son17, Zel14] or estimated algorithmically based on captured color images [Liu15, Mie18, Qin17, Weg18].

High-quality immersive views require a huge amount of data in the form of a dense set of views accompanied by high-quality depth maps. All of this data need to be transmitted over the network to a viewer's receiver in order to allow them to choose their own viewpoint of a scene.

In order to efficiently handle the multiview data, many compression technologies have been proposed. Most of them are based on the single-view video compression technology. Therefore, the multiview extension of the very popular AVC compression technology is developed as 3D-AVC [Ann14, Han13]. Similarly, the newer HEVC technique has its own multiview extension in the form of a 3D-HEVC codec [Ann18, Tec16]. Both of them have been developed almost at the same time around 2015 by the experts of the Joint Collaborative Team on 3D Video Coding Extension Development (JCT-3V) [JCT17].

https://doi.org/10.24132/CSRN.2020.3001.17

The main improvement over the single-view compression is achieved by extending the interframe motion-compensated prediction mechanism to interview disparity-compensated prediction. Owing to the improvement, as well as to some minor tools, multiview extensions allow a 30% bitrate reduction in comparison to simply coding all of the data in parallel by single-view codecs (simulcast coding) [Tec16].

Additionally, in 3D-HEVC, several tools designed especially for depth coding have been introduced. Some of them allow the prediction of depth data based on color pictures [Mul12], while others exploit specialized prediction of depth [Mul12, Tec16]. All of those tools allow further bitrate reduction. However, as depth data are not a huge component in the compressed data stream, any improvements in depth coding do not translate directly into overall coding performance improvement.

The 3D-HEVC was developed mainly for multi-camera systems with linear camera arrangement (Figure 1a). At the time of 3D-HEVC development, i.e., around 2011-2013, such applications were considered as important, e.g., with respect to the autostereoscopic display technology. Therefore, many tools in 3D-HEVC have been developed especially for linearly arranged views, and thus do not provide any compression gain for other types of multiview content [Sam16, Sta15].

Currently, many researchers focus on multiview systems with cameras located either around a scene (Figure 1b), in an idealized case on an arc around a scene. Such a setup allows better impressiveness, wider perspective change during free navigation and more freedom in choosing the view position, while it is not so difficult to build and calibrate such a camera system.

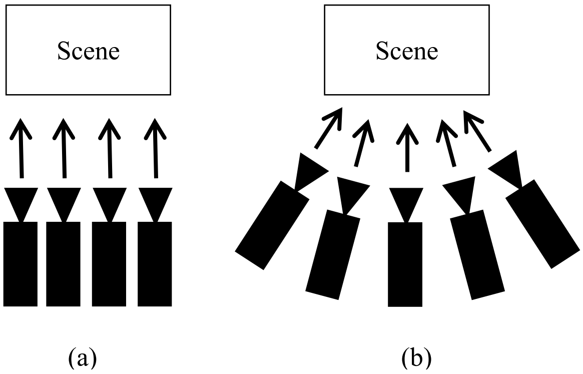


Figure 1. Camera arrangements: (a) linear (b) nonlinear

One of the most important problems with multiview compression stems from different characteristics of its components: color images (views) and depth data. Both can be compressed with different strength resulting from different overall characteristics. This problem can be stated differently: having the number of bits fixed, what number of bits should be allocated to color images (views) and to depth data in order to obtain the highest quality of a virtual view generated from decoded data at the decoder. When the compression of depth data is too strong, spatial relationships are not well-preserved and views synthesized based on the decoded data lose quality, due to spatial distortions. Similarly, too strong compression of color images makes the views generated at the decoder blurry and low in quality. Therefore, a “sweet spot”, where bits spent for both components of multiview data are balanced, must exist.

2. PREVIOUS WORKS

Many solutions for the best allocation of bits between color and depth data have already been found. In most of them, the encoder is controlled via setting quantization parameter for color images (*QP*) and depth maps (*QD*) separately. Therefore, many researchers focus on providing pairs of quantization parameters for color and depth data which lead to the best quality virtual views generated out of decoded data at the particular bitrate. Many such pairs can be found in the literature, but the optimum golden standard still remains an open question. In [Bos11, Bos13], the authors have estimated the optimal bitrate ratio based on only two multiview sequences but did not supply any formula for calculating the optimal quantization parameters for video and depth data. In [Kli14a, Kli14b], the authors have proposed formulas that permit optimal bitrate allocation in the multiview video plus depth compression based on only one quantization parameter (*QP*) and estimating the second one based on previously derived formulas. Regretfully, the authors did not report performance analysis (bitrate reduction), especially they did not compare the proposed model with manually selected optimum quantization pairs for the sequences used. In [Sta13a], the authors presented a model showing the relationship between the quantization parameter of the color data (*QP*) and depth data (*QD*) in 3D-AVC coding. But again, the model has not been compared to manually selected optimum pairs of both quantization parameters. The formulas for optimum bitrate allocation in multiview video compression plus depth using simulcast HEVC, simulcast VVC, and MV-HEVC codecs were derived in [Alo18b, Alo19]. Furthermore, bitrate reduction between the proposed model and coding with the straightforward approach keeping both quantization parameters equal (*QP=QD*) has been reported.

3. METHODOLOGY

In the experiments, we have simulated a simple FTV system where two views and two depth maps are transmitted and a viewer always selects in-between input views the virtual view position to be synthesized.

A block diagram of the simulated system has been presented in Figure 2. Videos and depth maps have been encoded and then fully decoded using 3D-HEVC. Next, the decoded views with associated depth maps have been used to construct a requested virtual view. The generated virtual view has been compared via PSNR of luminance with the view acquired by the real camera precisely in the same position in 3D space as the generated virtual view. Finally, the measured PSNR of the generated virtual view and total bitrate of the data necessary for constructing it, have been gathered and plotted on a chart.

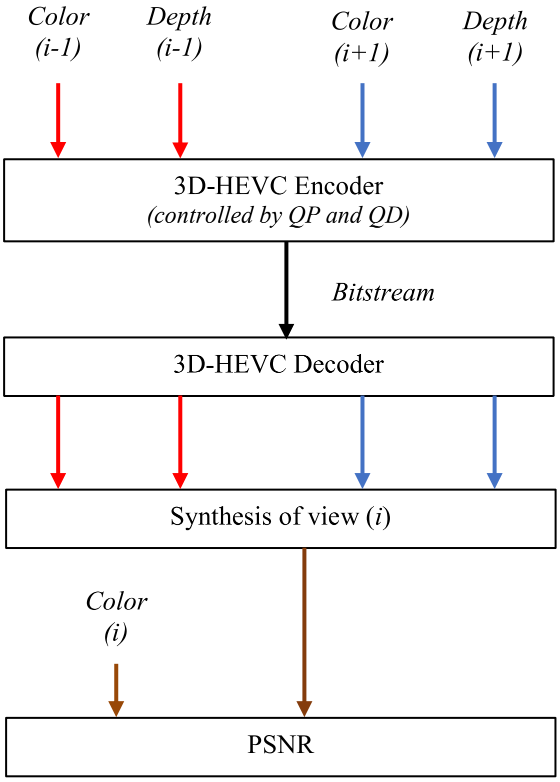


Figure 2. Structure of performed experiments

During the experiments we have examined (encode, decode, and synthesize) all of the possible pairs of quantization parameters in the range of 25 to 51. It results in 27 x 27 encodings and virtual views generated, for which we have gathered the data.

The experiments have been conducted on eight multiview sequences recommended by the Moving Picture Experts Group (MPEG) affiliated by the International Organization for Standardization (ISO). Moreover, all selected sequences have provided high-quality depth maps for all of the views.

In order to test our findings, we have divided the multiview sequences into two groups (Table 1). The first group is called a training group and was used to estimate the parameters of the proposed model. The second group is called the verification group, and was used to assess the performance of the proposed model.

As mentioned at the beginning, three views have been used for each sequence; two views have been used to produce a virtual view, while the third view has been used as a view synthesis reference for quality assessment. For the virtual views synthesis, state-of-the-art synthesis software called View Synthesis Reference Software [Sta13b] has been used. This software package developed by MPEG allows high-quality virtual view rendering based on two videos and two depth maps.

Sequence name	Resolution	Position of the encoded views	Position of the synthesized view
Training set			
Ballet [Zit04]	1024×768	3, 5	4
Breakdancers [Zit04]	1024×768	2, 4	3
BBB Butterfly [Kov15]	1280×768	49, 51	50
BBB Flowers [Kov15]	1280×768	39, 41	40
Kermit [Sal19]	1920×1080	5,7	6
Poznan_CarPark [Dom09]	1920×1088	3,5	4
Verification set			
Poznan_Block2 [Dom16]	1920×1080	2, 6	4
Poznan_Fencing [Dom16]	1920×1080	2, 6	4

Table 1. Test sequences used in experiments

For the experiments, version 3D-HTM 16.2 of the 3D-HEVC reference software [3DHEVC] was used. The encoder was configured according to the MPEG common test conditions for 3D video [Mul13].

To simplify and study bitrate allocation between views and depth maps, we have assumed that the quantization parameter for video (*QP*) is constant for all views, and the quantization parameter for depth maps (*QD*) is also fixed for all depth maps. Thus, two quantization parameters have been used to control the encoder instead of four.

4. THE PROPOSED MODEL

We have assumed that there exists a simple linear relationship between the quantization parameters for color data (*QP*) and depth data (*QD*).

$$QD(QP) = \alpha \cdot QP + \beta \qquad (1)$$

For estimating the parameters of the proposed model, first we need to find all optimum quantization parameter (*QP-QD*) pairs for multiview sequences from the training set. Similarly to [Alo18a, Alo19], we have tested all possible quantization parameter pairs (*QP-QD* pairs) in the range of 25 to 51.

In Figure 3, we have presented the obtained R-D (rate-distortion) curves. Each red point represents a coding result of one quantization pair (*QP-QD* pair). Let us bear in mind that the quality of compression is assessed as the quality of the virtual view generated based on decoded data. Based on this raw data we have selected all the pairs of quantization parameters (*QP-QD* pairs) which lead to the best quality of a virtual view generated out of data compressed at the given bitrate (blue lines in Figure 3). The optimum quantization pairs (*QP-QD* pairs) belong to an envelope of a raw cloud of PSNR-bitrate points. Those selected optimal pairs of quantization parameters have been presented onto a *QP-QD* plain (Figures 4 and 5). As it can be seen, those pairs lie almost on a straight line.

By means of linear regression, we have estimated the parameters of a linear model connecting the quantization parameter of depth data with the quantization parameter of color images. The estimated parameters have been gathered in Table 2. Our model allows us to calculate one quantization parameter based on the other, while maintaining the optimal bitrate allocation.

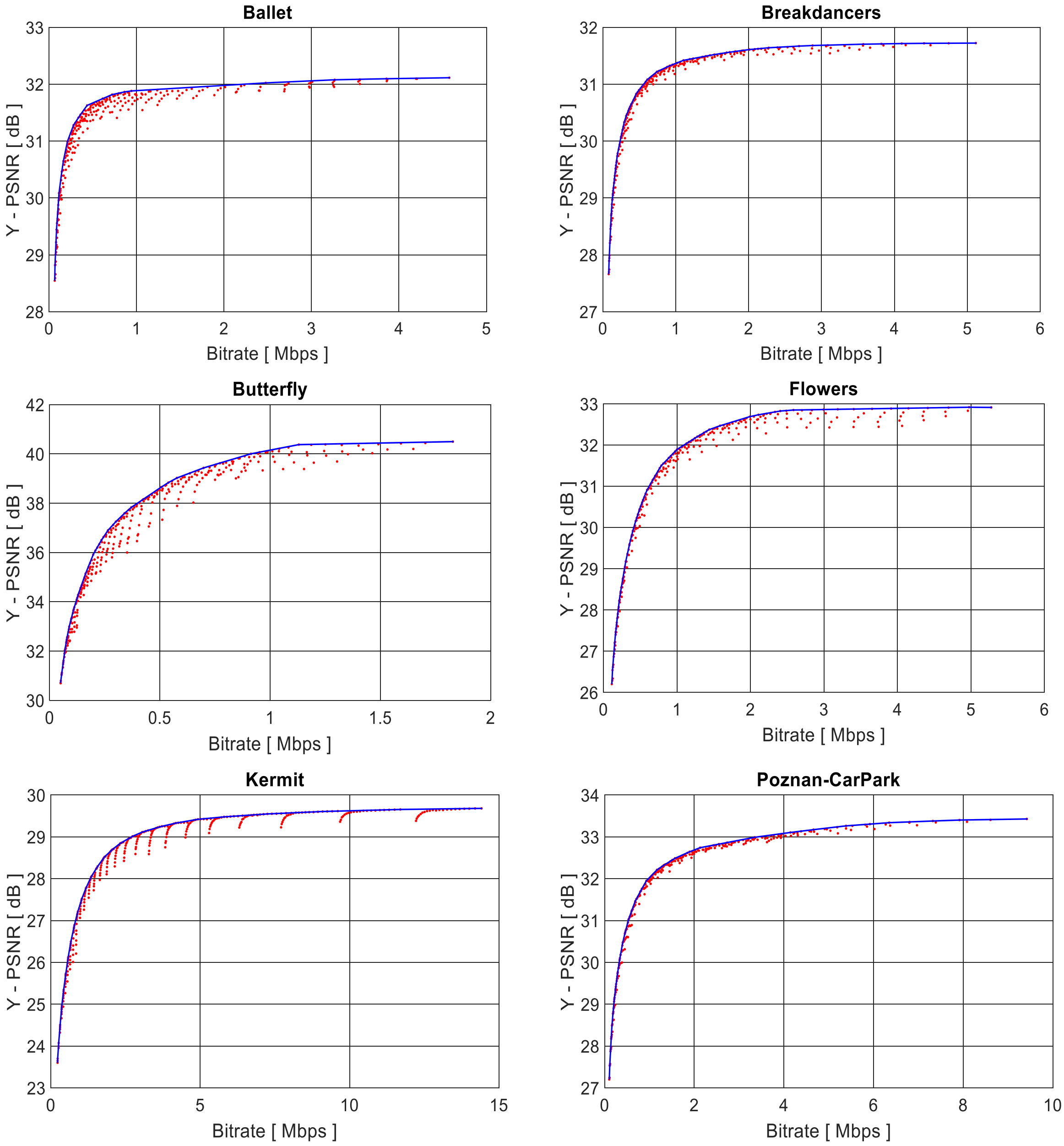


Figure 3. The best curve calculated by coding a video with all *QP-QD* pairs

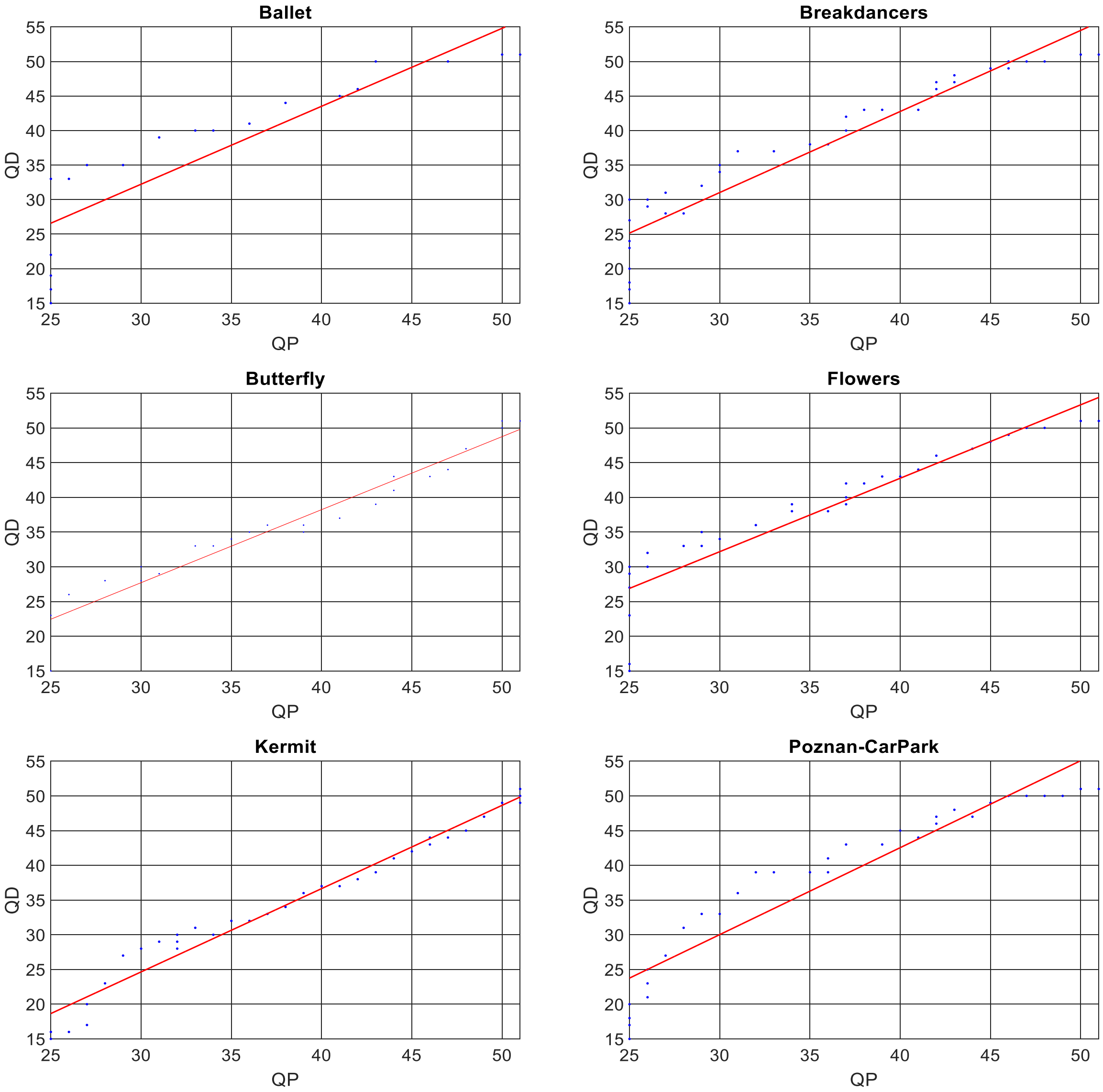


Figure 4. The approximate relationship between *QP* and *QD* for the optimum pairs for each training sequence with the use of linear regression

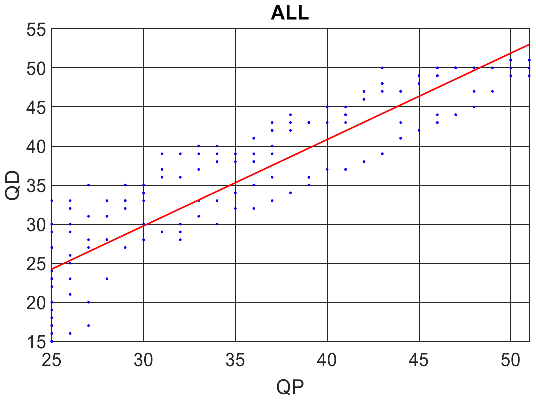


Figure 5. Optimum *QP-QD* pairs for all training sequences (blue points) and approximation line for *QP-QD* relationship (red line)

Sequence	*α*	*β*
Ballet	1.13	-1.64
Breakdancers	1.17	-4.19
Butterfly	1.05	-3.84
Flowers	1.06	-0.44
Poznan_CarPark	1.25	-7.55
Kermit	1.20	-11.34
Average	**1.11**	**-3.40**

Table 2. Parameters α and β for the linear regression model approximation for optimum *QP-QD* pairs.

5. PERFORMANCE OF THE PROPOSED MODEL

We have tested the proposed model in order to compare the optimum pairs of quantization parameters with another test set (verification set). As mentioned before, it was composed of two multiview sequences: Poznan_Block2 and Poznan_Fencing.

We have encoded both sequences with quantization parameters according to our proposed model (according to average values from Table 2) and with the quantization parameter pairs provided in the common test condition (CTC) document used during the development of 3D-HEVC [Mul13]. CTC specifies in detail the encoder configuration, including quantization parameter pairs for 3D-HEVC testing.

After the encoding, we have compared the total resulting bitrate and the quality of the virtual view generated based on the decoded data.

The comparison of the proposed model with the CTC conditions has been performed by calculating the average difference between the curves for PSNR (ΔPSNR) and bitrate (ΔBitrate). The calculated ΔPSNR and ΔBitrate are just a simple extension of the well-known Bjøntegaard metric [Bjo01] to work with more than four points. The obtained results can be found in Tables 3 and 4. Figure 6 presents R-D (rate-distortion) curves for one of the verification multiview sequences - Poznan_Block2.

	Tested Reference	CTC	Proposed model	Optimum *QP-QD* pairs
Poznan_ Block2	CTC	N/A	**-34.36%**	-51.99%
	Model	52.35%	N/A	-27.10%
	Optimum	108.29%	37.17%	N/A
Poznan_ Fencing	CTC	N/A	**-23.05%**	-44.44%
	Model	31.58%	N/A	-28.22%
	Optimum	79.99%	40.26%	N/A

Table 3. Comparison of Bjøntegaard ΔBitrate metrics for optimum *QP-QD* pairs, common test condition (CTC) and the proposed model for 3D-HEVC

Experiments conducted on a verification set have shown that the proposed model for calculating quantization parameter pairs (Table 3) led to a decrease of total bitrate and improved the virtual view quality of sequences when compared to the quantization pairs recommended in the common test condition (CTC) document (23-35%). However, as always in modeling, the usage of the model leads to an increase in total bitrate and a decrease of virtual view quality in comparison to the usage of optimum *QD-QP* pairs. It is worth noticing that optimum *QP-QD* pairs for a given sequence are not known in advance, especially when new content is used. This is why the proposed model is an improvement over the recommendation included in the CTC document.

	Tested Reference	CTC	Proposed model	Optimum *QP-QD* pairs
Poznan_ Block2	CTC	N/A	**0.28 dB**	0.51 dB
	Model	-0.28 dB	N/A	0.22 dB
	Optimum	-0.51 dB	-0.22 dB	N/A
Poznan_ Fencing	CTC	N/A	**0.05 dB**	0.12 dB
	Model	-0.05 dB	N/A	0.06 dB
	Optimum	-0.12 dB	-0.06 dB	N/A

Table 4. Comparison of Bjøntegaard ΔPSNR metrics for optimum *QP-QD* pairs, common test condition (CTC) and the proposed model for 3D-HEVC

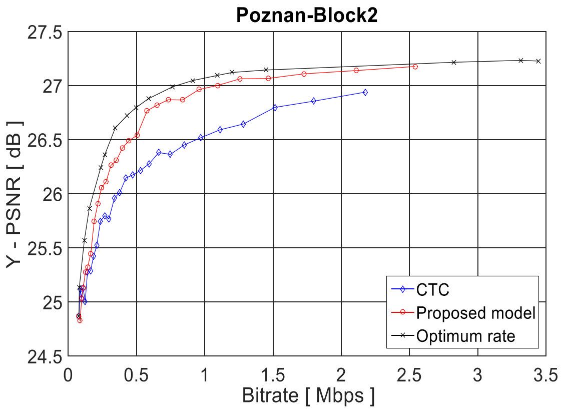


Figure 6. R-D curves comparison between the proposed model, CTC approach, and optimum *QP-QD* pairs for 3D-HEVC coding for the Poznan_Block2 sequence

6. CONCLUSIONS

In the paper, we have proposed a simple model for optimal bitrate allocation between color and depth data in 3D-HEVC coding. The provided model allows better bitrate division between color and depth data, which leads to a significant (23-35%) bitrate reduction of the total bitrate of the multiview stream. At the same time it preserves the same quality of synthesized virtual views in comparison to the common test condition (CTC) recommendation, which is considered as a well-established reference encoder configuration.

Moreover, based on the proposed model, we can control multiview compression by using only one quantization parameter instead of two parameters.

7. ACKNOWLEDGMENTS

The research was supported by the Ministry of Science and Higher Education of the Republic of Poland.

8. REFERENCES

[Alo18a] Y. Al-Obaidi, T. Grajek, “Influence of depth map fidelity on virtual view quality,” Int. Conf. on Signals and Electronic Systems (ICSES), Kraków, Poland, Sep. 2018.

[Alo18b] Y. Al-Obaidi, T. Grajek, O. Stankiewicz, M. Domanski, “Bitrate allocation for multiview video plus depth simulcast coding,” Int. Conf. Systems, Signals, and Image Proc. (IWSSIP), Maribor, Slovenia, 2018.

[Alo19] Y. Al-Obaidi, T. Grajek, M. Domanski, “Quantization of Depth in Simulcast and Multiview Coding of Stereoscopic Video plus Depth Using HEVC, VVC and MV-HEVC,” Picture Coding Symposium (PCS), Ningbo, China, Nov. 2019.

[Ann14] Annex I “Multiview and Depth video coding” of ISO/IEC 14496-10, Int. Standard “Generic coding of audio-visual objects – Part 10: Advanced Video Coding,” 8th Ed., 2014, also: ITU-T Rec. H.264, Edition 12.0, 2017.

[Ann18] Annex I “3D high efficiency video coding” of ISO/IEC 23008-2, Int. Standard “High efficiency coding and media delivery in heterogeneous environments - Part 2: High efficiency video coding,” 4th Ed., 2018, also: ITU-T Rec. H.264, Edition 7.0, 2019.

[Bjo01] G. Bjontegaard, “Calculation of average PSNR differences between RD-curves,” ITU-T SG16, Doc. VCEG-M33, USA, 2001.

[Bos11] E. Bosc, V. Jantet, M. Pressigout, L. Morin, C. Guillemot, “Bit-rate allocation for multi-view video plus depth,” 3DTV Conf.: The True Vision -Capture, Transmission, and Display of 3D Video (3DTV-CON), Antalya, Turkey, June 2011.

[Bos13] E. Bosc, F. Racape, V. Jantet, P. Riou, M. Pressigout, L. Morin, “A study of depth/texture bit-rate allocation in multi-view video plus depth compression,” Annals of telecommunications, Vol. 68, Issue 11–12, pp 615–625, 2013.

[Dom09] M. Domański, T. Grajek, K. Klimaszewski, M. Kurc, O. Stankiewicz, J. Stankowski, K. Wegner, “Poznań multiview video test sequences and camera parameters,” ISO/IEC JTC1/SC29/WG11 MPEG2009, M17050, Xi'an, China, Oct. 2009.

[Dom16] M. Domański, A. Dziembowski, A. Grzelka, D. Mieloch, O. Stankiewicz, K. Wegner, “Multiview test video sequences for free navigation exploration obtained using pairs of cameras,” ISO/IEC JTC1/SC29/WG11, Doc. MPEG M38247, Geneva, Switzerland, 2016.

[Feh04] C. Fehn, “Depth-image-based rendering (DIBR), compression, and transmission for a new approach on 3D-TV,” Proc. SPIE, 2004.

[Han13] M. Hannuksela, D. Rusanovskyy, W. Su, L. Chen, R. Li, P. Aflaki, D. Lan, M. Joachimiak, H. Li, M. Gabbouj, “Multiview-video-plus-depth coding based on the advanced video coding standard,” IEEE Trans. Image Proc., Sep. 2013.

[HEVC] 3D HEVC reference codec available online https://hevc.hhi.fraunhofer.de/svn/svn_3DVCSoftware/tags/HTM-16.3

[Isg04] F. Isgro, E. Trucco, P. Kauff, O. Schreer, “Three-dimensional image processing in the future of immersive media,” IEEE Trans Circuits Syst. Video Techn., vol. 14, pp. 288 – 303, 2004.

[JCT17] JCT-3V. JCT-3V Document Management System, Sep. 2017.

Accessed: http://phenix.int-evry.fr/jct2

[Kli14a] K.Klimaszewski, K.Wegner, M.Domański, “Video and Depth Bitrate Allocation in Multiview Compression,” Int. Conf. Systems, Signals, and Image Proc. (IWSSIP), Dubrovnik, Croatia, June 2014.

[Kli14b] K. Klimaszewski, O. Stankiewicz, K. Wegner, M. Domański, “Quantization optimization in multiview plus depth video coding,” IEEE Int. Conf. on Image Proc. (ICIP), Paris, France, Oct. 2014.

[Kov15] P. Kovacs, “[FTV AHG] Big Buck Bunny light-field test sequences,” ISO/IEC JTC1/SC29/WG11, Doc. MPEG M35721, Geneva, Switzerland, 2015.

[Mie18] D. Mieloch, A. Grzelka, “Segmentation-based method of increasing the depth maps temporal consistency,” Int. Journal of Electronics and Telecommunications, Warsaw, Poland, 2018.

[Liu15] C. Liu, W. Zhang, Z. Qi, L. Shi, “A robust temporal depth enhancement method for dynamic virtual view synthesis,” 23rd International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, WSCG 2015 - Full Papers Proceedings, pp. 191-200, Plzen, Czech Republic, 2015.

[Mul11] K. Muller, P. Merkle, T. Wiegand, “3-D video representation using depth maps,” Proc. IEEE, April 2011.

[Mul12] K. Müller, P. Merkle, G, Tech, T, Wiegand, “3D video coding with depth modeling modes and view synthesis optimization,” Proceedings of The 2012 Asia Pacific Signal and Information

Processing Association Annual Summit and Conference, Hollywood, USA, 2012

[Mul13] K. Müller, A. Vetro, "Common test conditions of 3DV core experiments," ISO/IEC JTC1 SC29/WG11 and ITU-T SG 16 WP 3, Doc. JCT3V G1100, San José, Jan. 2014.

[Par20] K. Park, S. Kim, K. Sohn, "High-precision depth estimation using uncalibrated LiDAR and stereo fusion," IEEE Trans. Intelligent Transportation Systems, Jan 2020.

[Qin17] Y. Qin, X. Jin, Y. Chen, Q. Dai, "Enhanced depth estimation for hand-held light field cameras," IEEE Int. Conf. Acoustics, Speech, and Signal Proc. (ICASSP), New Orleans, LA, 2017.

[Sal19] B. Salahieh, B. Kroon, J. Jung, M. Domański, "Test Model for Immersive Video," ISO/IEC JTC1/SC29/WG11 MPEG/N18470, Geneva, Switzerland, April 2019.

[Sam16] J. Samelak, J. Stankowski, M. Domański, "Adaptation of the 3D-HEVC coding tools to arbitrary locations of cameras," Int. Conf. on Signals and Electronic Systems (ICSES), Kraków, Poland, Sep. 2016.

[Son17] Y. Song, Y. Ho, "High-resolution depth map generator for 3D video applications using time-of-flight cameras," IEEE Trans. Consumer Electronics, vol. 63, Nov. 2017.

[Sta13a] O. Stankiewicz, K. Wegner, M. Domański, "AHG14: Optimized QP/QD curve for 3D coding with half and full resolution depth maps," ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, Doc. JCT-3V E0269, Vienna, Austria, Aug. 2013.

[Sta13b] O. Stankiewicz, K. Wegner, M. Tanimoto, M. Domański, "Enhanced View Synthesis Reference Software (VSRS) for Free-viewpoint Television," ISO/IEC JTC 1/SC 29/WG11, Doc. M31520, Geneva, Switzerland, 2013.

[Sta15] J. Stankowski, Ł. Kowalski, J. Samelak, M. Domański, T. Grajek, K. Wegner, "3D-HEVC extension for circular camera arrangements," 3DTV Conference: The True Vision-Capture, Transmission and Display of 3D Video (3DTV-Con 2015), Lisbon, Portugal, Jul. 2015.

[Sta18] O. Stankiewicz, M. Domański, A. Dziembowski, A. Grzelka, D. Mieloch, J. Samelak, "A free-viewpoint television system for horizontal virtual navigation," IEEE Trans. Multimedia, Jan. 2018.

[Tan12] M. Tanimoto, M. Panahpour, T. Fujii, T. Yendo, "FTV for 3 D spatial communication," Proc. IEEE, Feb. 2012.

[Tec16] G. Tech, Y. Chen, K. Müller, J. Ohm, A. Vetro, Y.Wang, "Overview of the multiview and 3D extensions of high efficiency video coding," IEEE Trans. Circuits Sys. Video Technology, Jan. 2016.

[Weg18] K. Wegner, O. Stankiewicz, T. Grajek, M. Domański, "Depth estimation from stereoscopic 360-degree video," IEEE Int. Conf. Image Proc. (ICIP), Athens, Greece, Oct. 2018.

[Zel14] N. Zeller, F. Quint, L. Guan, "Kinect based 3D scene reconstruction," 22nd International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, WSCG 2014, pp. 73-81, Plzen, Czech Republic, 2014.

[Zin10] S. Zinger, L. Do, P. H. N. de With, "Free-viewpoint depth image based rendering," J. Vis. Commun. Image Represent., vol. 21, no. 5, pp. 533–541, Jan. 2010.

[Zit04] L. Zitnick, S.B. Kang, M. Uyttendaele, S. Winder, R. Szeliski, "High-quality video view interpolation using a layered representation," ACM SIGGRAPH, pp. 600-608, 2004.

Pose and Visual Attention: Exploring the Effects of 3D Shape Near-Isometric Deformations on Gaze

Valeria Garro
Blekinge Institute of Technology
371 79 Karlskrona, Sweden
valeria.garro@bth.se

Veronica Sundstedt
Blekinge Institute of Technology
371 79 Karlskrona, Sweden
veronica.sundstedt@bth.se

ABSTRACT

Recent research in 3D shape analysis focuses on the study of visual attention on rendered 3D shapes investigating the impact of different factors such as material, illumination, and camera movements. In this paper, we analyze how the pose of a deformable shape affects visual attention. We describe an eye-tracking experiment that studied the influence of different poses of non-rigid 3D shapes on visual attention. The subjects free-viewed a set of 3D shapes rendered in different poses and from different camera views. The fixation maps obtained by the aggregated gaze data were projected onto the 3D shapes and compared at vertex level. The results indicate an impact of the pose for some of the tested shapes and also that view variation influences visual attention. The qualitative analysis of the 3D fixation maps shows high visual focus on the facial regions regardless of the pose, coherent with previous works. The visual attention variation between poses appears to correspond to geometric salient features and semantically salient parts linked to the action represented by the pose.

Keywords

Shape Analysis, Mesh Models, Visual Attention, Perception, Gaze Analysis.

1 INTRODUCTION

Understanding and modeling human visual attention is a relevant research topic that has been widely investigated in computer graphics and computer vision [Bor13]. Several applications relying on user's gaze detection and analysis have been proposed in the last decades. Foveated or selective rendering [Pat16] and mesh simplification [Lar11] are some examples of computer graphics applications aiming at high perceived visual quality of the scene. In computer vision, many object recognition and object detection applications are based on saliency models [Bor13].

In 3D shape analysis, the concept of *mesh saliency* was introduced in [Lee05]. The authors presented a method to compute a per-vertex saliency measure on the 3D mesh inspired by low-level visual attention. Several other works, e.g. [Wu13, Lei16], proposed similar surface-based saliency models focusing on the geometric properties of the 3D mesh. Recent works provided a step forward into the analysis of visual attention applied to static 3D shapes, investigating the influence of other factors such as different camera views [McD09], rendering lighting conditions, materials, and camera movements [Lav18]. In general, the datasets used in these works consist of rigid 3D shapes such as 3D models of statues, vases, mechanical objects, and non-rigid shapes in resting positions like a quadruped animal standing on four legs. However, in real application scenarios, it is common to deal with deformable shapes that move and assume different poses, in addition to views. For this reason, we focus on investigating visual attention variations when looking at the same shape holding different poses. Our contribution with this paper is to perform a first step to explore how pose could potentially influence visual attention. This is done by acquiring and analyzing eye-tracking data from observers viewing near-isometric deformations of 3D shapes, i.e. human or animal shapes, in different poses.

In the experiment presented in this paper, we asked the participants to free-view a computer screen displaying a set of 3D shapes rendered in different poses and from different angles. We acquired the participants' gaze data using an eye tracker and we computed the 2D aggregated fixation maps. Each 2D fixation map has then been projected onto the related 3D shape transferring the fixation data to the vertices of the mesh and creating a *3D fixation map*. The 3D fixation maps of the same shape in different poses have then been compared to each other to analyze variations in visual attention.

https://doi.org/10.24132/CSRN.2020.3001.18

The remainder of the paper is organized as follows: Section 2 presents previous works on visual attention related to 3D models by eye-tracking analysis; Sections 3 and 4 describe the process to create the stimuli used during the experiment, the equipment used, and the experiment methodology. Section 5 presents the process of creating the 3D fixation maps, while the comparison between 3D fixation maps is described in Section 6. Since we aggregate the fixation data from our valid set of participants, an analysis of the consistency across participants' data (Inter Observer Congruency) is required and it is described in Section 7. The results are analyzed in Section 8, while in Section 9 we summarize the findings of our work.

2 RELATED WORK

In the last decades, several works investigated visual attention on 3D objects through the analysis of eye-tracking data focusing on different perspectives. In [How05], the authors, through a set of experiments differing in tasks, analyzed gaze data to determine salient features of 3D models in the context of mesh simplification to maintain high perceptual quality. The sets of 3D models used in these experiments include natural objects, i.e. animal shapes, and artifacts. One of the results showed that the heads of natural objects obtained high values of saliency. Similar findings were presented in [McD09]; this work focused on the analysis of salient body parts of virtual human characters in crowd rendering, revealing high saliency for the heads and the upper torsos. The models displayed in these experiments were textured human characters wearing different casual outfits, either standing in a neutral position or performing walk cycles.

In [Kim10], the mesh saliency model presented by Lee et al. [Lee05] was compared with fixation data acquired through an experiment to verify their similarity. In this experiment, Lee et al. saliency model showed higher correlation values than a random model indicating a correspondence between the saliency model prediction and human eye fixations. Here, the comparison analysis was performed at image level applying a modified version of the chance-adjusted saliency metric [Par02]. More recent works studied visual attention directly on the 3D shapes, both real 3D printed [Wan16, Wan18] and virtual [Lav18]. Saliency models like [Lee05] were also analyzed in [Wan16, Lav18], however in these studies they exhibited poor performance in predicting human fixations, demonstrating the complexity of the visual attention process.

These existing saliency models are based only on the analysis of the geometric features of the shapes. The authors of [Lav18, Wan18] studied also the impact of other conditions on visual attention, such as view orientation and material. In [Lav18], it was reported a significant influence of material, light setting, and camera paths on fixation data when inspecting virtual 3D models. In a setting with real objects [Wan18], different camera views provided different results on fixations; in contrast, the two analyzed materials did not determine any significant difference. Based on these recent findings, we were inspired to go a step further and analyze a property that, to the best of our knowledge, has not studied before, the impact of different poses of non-rigid shapes from different views.

3 CREATION OF THE STIMULI

Six non-rigid characters have been selected from the TOSCA high-resolution dataset [Tos, Bro08] for non-rigid shape similarity and correspondence experiments. Meshes of the same character in different poses have the same triangulation and the vertices are listed in the same order. This allows a direct comparison at a per-vertex level. The selected characters are two human males, a horse, a cat, a gorilla, and a centaur. The average number of vertices is about 35000. For each of the chosen characters, four different poses have been selected for a total of 24 different meshes. As shown in Figure 5 and Figure 6, the poses include resting positions (e.g. *Horse0*, *Michael15*), as well as extreme actions (e.g. *Horse10*, *Centaur3*), common (e.g. *Cat2*) and more uncommon (e.g. *Gorilla8*) positions.

Each mesh was rendered from three different camera locations: one in front of the mesh (V_0), one at a 45° angle (V_{45}), and the last one at 90° angle (V_{90}). For meshes belonging to the same character, the cameras were positioned at a constant distance from the center of mass of each mesh. The height of the camera center was set at the same height as the center of mass of the mesh. All meshes were rendered using Blender 2.80[1] Eevee engine, with a Lambertian gray shading. The same lighting setup was used for each camera view, with a single light providing uniform illumination positioned behind the camera and pointing at the same direction of the camera. The final rendered images have a resolution of 1920×1080.

4 EXPERIMENT EQUIPMENT AND PROCEDURE

The stimuli were visualized on a 15.6 inches laptop screen. A Tobii X2-30 eye tracker was used to collect the gaze data of the participants. This device has a sampling rate of 30 Hz and a gaze accuracy of 0.4° under ideal conditions. Tobii Pro Studio software[2] was used to design the experiment and record the gaze data. 21 participants were recruited for the experiment, having normal or corrected-to-normal vision and they were

[1] https://www.blender.org

[2] https://www.tobiipro.com/product-listing/tobii-pro-studio/

not aware of the purpose of the experiment. Participants' age ranged from 19 to 46. The reported genders were 7 F, 13 M, and 1 "Prefer not to answer". The data collected from four participants were rejected due to calibration issues or a low percentage of valid detected gaze. After reading the experiment instructions and signing the consent form, the session started with a 9-points calibration procedure for the eye-tracker device. During the 9-points calibration, the participant was asked to follow a moving red dot on the screen, this procedure lasts for around 40 seconds. As an additional check of the calibration accuracy, the first stimuli set were four images of a cross positioned in different areas of the screen. Each image was shown for four seconds and the participants were asked to look at the center of each cross. Thereafter, the participants were asked to free-view the set of 72 rendered images, each for seven seconds with a monochromatic (mid-gray) image in between the stimuli of the duration of two seconds to separate stimuli reactions. The order in which the rendered images were displayed was randomized for each participant to prevent an order effect bias [Cun11].

5 3D FIXATION MAPS

The first step for the creation of the 3D fixation maps is collecting the aggregated gaze data for each stimulus. The identification of fixations has been performed applying the I-VT fixation filter provided by Tobii which classifies eye movements based on their velocity [Kom10]. For each mesh C_i, representing the character C in pose i, and each view j, the set of fixations of all participants were aggregated in a 2D fixation map f_{ij}^C, a 2D matrix of the same size of the stimuli. Each fixation contributes by adding to f_{ij}^C a two-dimensional gaussian kernel centered on the pixel coordinates of the fixation point with a radius of 62 pixels which corresponds in our setup to a visual angle of 1°, i.e. the radius of the fovea, the region in the visual field with highest visual acuity [Duc17]. The values of the 2D fixation map are mapped to the unit interval creating a grayscale image.

We project the 2D fixation image on the related 3D mesh with Meshlab software [Mes]. The current 3D mesh has now grayscale color values at vertex level representing the fixation values, with bright color values corresponding to areas related to high fixation values and dark color values indicating low or absent fixations. The 3D fixation map of the character C in pose i from view j is defined as the list of vertex color values. Since the shapes belong to the TOSCA high-resolution dataset, 3D meshes of the same character have the vertices listed in the same order. Hence, 3D fixation maps of the same character can be directly compared with a chosen similarity or distance metric.

6 3D FIXATION MAPS COMPARISON

Since the goal is to analyze the effects on visual attention of different poses of the same character, we compared all pairs of 3D fixation maps of the same character C obtained from the same view j. Pearson's Correlation Coefficient (PCC) is used as a comparison measure. For two 3D fixation maps X and Y, PCC is defined as $\text{cov}(X,Y)/(\sigma_X \sigma_Y)$, where cov indicates the covariance and σ the standard deviation. PCC estimates the linear relationship between two 3D fixation maps and its values range between -1 and 1, with 0 implying no correlation and 1 and -1 perfect positive and negative linear relationship respectively. PCC is an evaluation metric commonly applied to compute the similarity between 2D saliency maps [Byl19] and it has also been used in recent works on fixation maps applied to 3D shapes [Lav18].

We define V_{aj}^C the set of vertices of the mesh C_a visible from view j. When comparing the 3D fixation maps F_{aj}^C and F_{bj}^C related to meshes C_a and C_b, PCC is computed exclusively on $V_{aj}^C \cap V_{bj}^C$, i.e. the subset of vertices that are visible from view j for both meshes C_a and C_b, to assure the comparison is run solely on valid fixation values.

Due to the variability of the poses, it is unlikely that the observer will look at the exact same portion of the mesh when viewing two different poses of the same character. For this reason, in addition to PCC, the Jaccard Index J (intersection over union) of the two sets of visible vertices V_{aj}^C and V_{bj}^C is computed to measure their similarity: $J = |V_{aj}^C \cap V_{bj}^C| / |V_{aj}^C \cup V_{bj}^C|$. J values range from 0 (empty intersection) to 1 (equal sets). In this context, since PCC is computed on the intersection set, J indicates a measure of the extent of this common region on which the corresponding PCC was measured in relation to the union set of visible vertices. J indicates also how much the two poses vary from each other. If J is close to 0, it means that the two poses share a small set of vertices visible from the same view, due for example to auto-occlusions (e.g. the face of *Gorilla8* is hidden by the left arm from view V_{90}) or different nature of the poses (e.g. from view V_0, the abdomen of the cat character is visible in *Cat1* but completely hidden in *Cat0*), hence the observer looks overall at different portions of the same mesh. While if J is close to 1, it means that the sets of visible vertices of the two poses are almost congruent, hence the observer looks overall at a similar portion of the mesh.

7 INTER OBSERVER CONGRUENCY ANALYSIS

The similarity values obtained from the 3D fixation maps comparison rely on the aggregated gaze data gathered from all valid participants ($N = 17$). To investigate

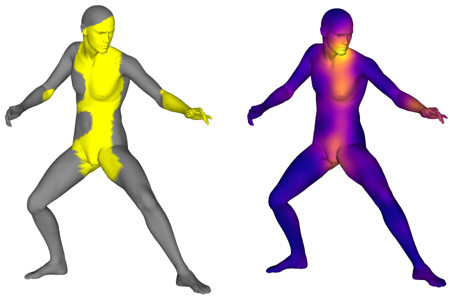


Figure 1: On the left, an example of binary fixation map with threshold 0.2 for *David10* obtained from view V_{45}. On the right, the corresponding original 3D fixation map color coded for visualization purpose, with yellow representing the highest number of fixations and blue indicating areas with no fixations.

the *Inter Observer Congruency* (IOC), i.e. the consistency across participants data, we adopt a *leave one out* approach similar to [Tor06]. For each stimulus, the congruency of the 3D fixation map generated from the data of one subject is tested against the partial aggregated 3D fixation map obtained by all the other $N-1$ participants. The final IOC value is generated by averaging the congruency values obtained with this procedure from all subjects.

A binary map indicating the most fixated vertices is created by setting a threshold to the partial aggregated 3D fixation map of $N-1$ participants. The vertices with fixation value > 0.2 have been selected creating the binary map which covers most of the aggregated fixations, as shown in Figure 1.

The fixations obtained by the left-out subject are then projected on the 3D mesh as circular patches to take into account eye tracker accuracy errors. The congruency value of the left-out subject is computed as the ratio of the vertices touched by a single observer fixations that fall also within the binary map.

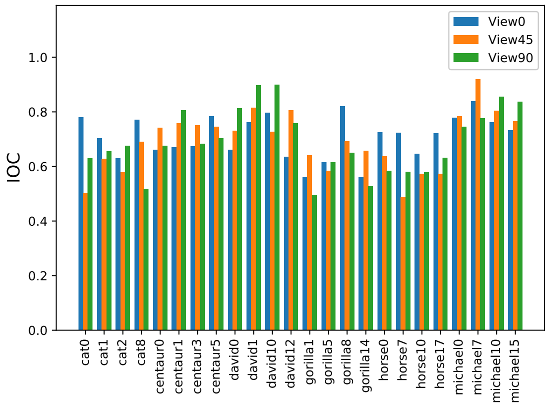


Figure 2: Inter Observer Congruency (IOC) mean values of all stimuli.

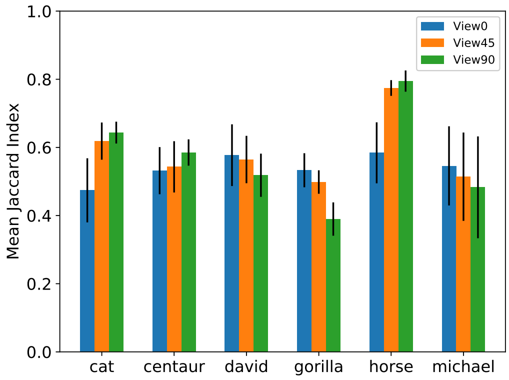


Figure 3: Mean and standard deviation values of Jaccard Index between pairs of shapes grouped by characters and views.

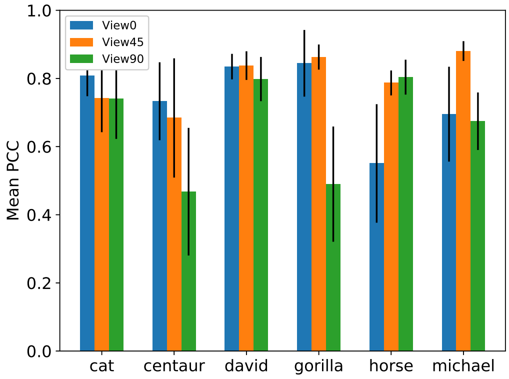


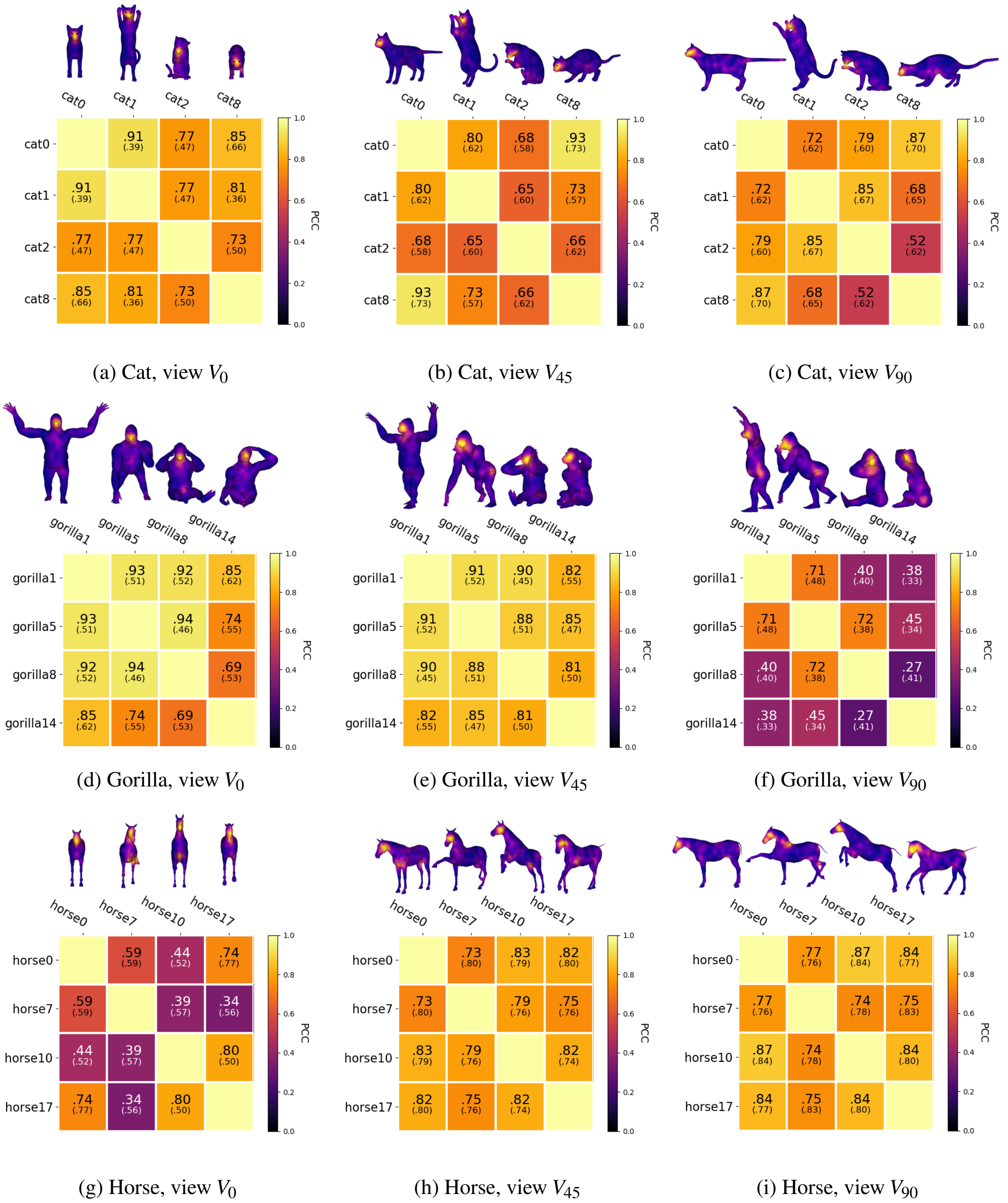
Figure 4: Mean and standard deviation values of Pearson's correlation coefficient grouped by characters and views.

8 RESULTS

The IOC results show a total mean value of 0.70 characterizing a fairly coherent dataset of fixations. Hence, the 3D fixation maps computed from aggregating the fixations data of all valid participants can be used on the comparison analysis.

A further analysis of the IOC values show a small variation between views ($\text{IOC}_{V_0} = 0.71$, $\text{IOC}_{V_{45}} = 0.69$, and $\text{IOC}_{V_{90}} = 0.69$) and a wider variation is presented when computing the mean IOC values of the different characters regardless of the view ($\text{IOC}_{\text{gorilla}} = 0.62$, $\text{IOC}_{\text{horse}} = 0.62$, $\text{IOC}_{\text{cat}} = 0.65$, $\text{IOC}_{\text{centaur}} = 0.71$, $\text{IOC}_{\text{david}} = 0.78$, $\text{IOC}_{\text{michael}} = 0.80$). Interestingly, the highest IOC values are the ones related to human characters showing higher consistency across participants when looking at humans. The mean IOC value for each stimulus is shown in Figure 2.

A first qualitative analysis of the 3D fixation maps supports previous findings [How05, McD09] of high visual attention values, across all three views, over characters' heads and human characters' torsos, as shown in Figure 5 and Figure 6. This seems to happen also if changing the pose.



(a) Cat, view V_0

(b) Cat, view V_{45}

(c) Cat, view V_{90}

(d) Gorilla, view V_0

(e) Gorilla, view V_{45}

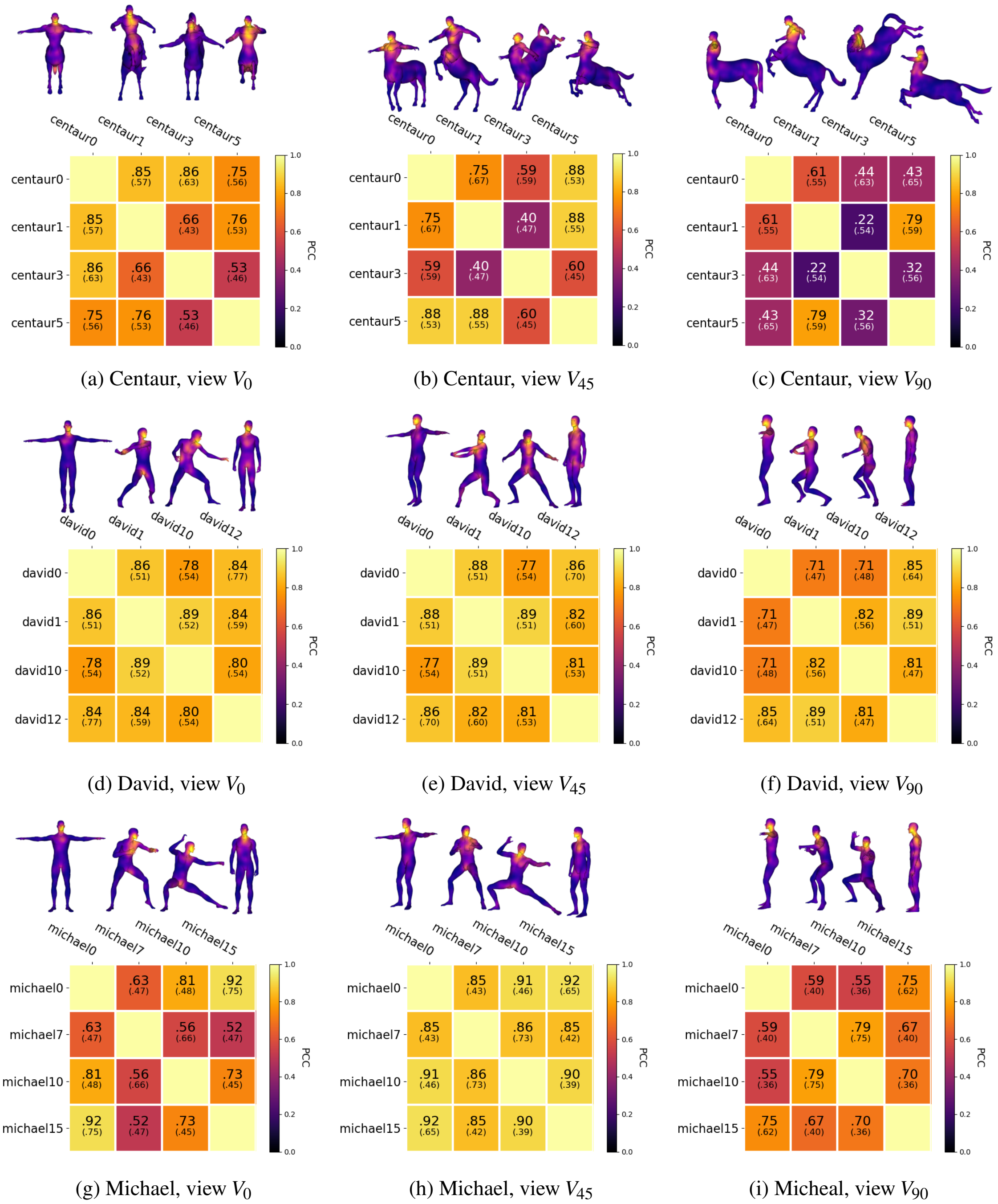
(f) Gorilla, view V_{90}

(g) Horse, view V_0

(h) Horse, view V_{45}

(i) Horse, view V_{90}

Figure 5: Similarity matrices for characters: *Cat*, *Gorilla*, and *Horse*. The similarity matrices show the Pearson's Linear Correlation (PCC) values between pairs of shapes belonging to the same character and looked from the same view. The corresponding Jaccard Index is indicated between parentheses. The aggregated 3D fixation maps related to each pose are shown above each similarity matrix. The 3D fixation maps are color coded for visualization purpose, with yellow representing the highest number of fixations and blue indicating areas with no fixations.



(a) Centaur, view V_0

(b) Centaur, view V_{45}

(c) Centaur, view V_{90}

(d) David, view V_0

(e) David, view V_{45}

(f) David, view V_{90}

(g) Michael, view V_0

(h) Michael, view V_{45}

(i) Micheal, view V_{90}

Figure 6: Similarity matrices for characters: *Centaur*, *David*, and *Michael*. The similarity matrices show the Pearson's Linear Correlation (PCC) values between pairs of shapes belonging to the same character and looked from the same view. The corresponding Jaccard Index is indicated between parentheses. The aggregated 3D fixation maps related to each pose are shown above each similarity matrix. The 3D fixation maps are color coded for visualization purpose, with yellow representing the highest number of fixations and blue indicating areas with no fixations.

Figure 3 shows the mean and standard deviation of the Jaccard Index values (intersection over union) computed from pairs of different poses of the same character. These values express a measure of variability of visible vertices between a pair of poses. Since PCC is computed on the intersection of the sets of visible vertices, the Jaccard Index measures also the extent of the portion of the mesh on which the corresponding PCC value was measured in relation to the union of the visible vertices. For example, the mean Jaccard Index for the gorilla shapes viewed from view V_{90} is 0.39 which implies that on average the pairs of these 3D meshes share only about two-fifths of the union of the visible vertices from view V_{90} due to variability in the poses. While the mean Jaccard Index value for the horse shapes viewed from view V_{90} is 0.79 which indicates that on average the pairs of these 3D meshes share about four-fifths of the union of the visible vertices, denoting that very similar portions of the mesh are visible from that view.

All the computed PCC resulted in positive values, hence we treat the data as similarity values between 0 and 1. Figure 4 shows the mean and standard deviation values obtained by pairs of 3D fixation maps belonging to the same character and view.

For some of the characters, the mean PCC values show a low similarity indicating that a different pose might influence visual attention. A variation of mean values of the character between views reveals a notable impact also of the view itself. This trend is not constant between characters, revealing that some poses seen from specific views influence the visual attention more than others, as shown by the similarity matrices of each character and view in Figures 5 and 6.

Three examples of low similarity values can be found in the *Gorilla* and *Centaur* characters from view V_{90} (Figures 5f and 6c) and the *Horse* character from view V_0 (Figure 5g). The corresponding 3D fixation maps show high fixation values over geometric salient features such as folds on the mesh (e.g. the fold between torso and leg in *Gorilla14*) as well as parts of the mesh related to the action the shape could represent, e.g. the front leg up in the air for *Horse7*.

Further examples indicating the influence of action representation can be found comparing the 3D fixation maps of specific pairs of poses. *Cat2* pose presents a cat licking its right paw, while *Cat8* shows a cat lurking. In Figures 5b and 5c, the 3D fixation maps of *Cat2* indicate considerable higher fixation values on the right leg compared to the 3D fixation maps of *Cat8*. This could be related to the proximity of the leg to the head of the cat or to the actual relevance of the leg in the context of the action that is represented. In Figure 6b, *Centaur1* and *Centaur3* show two opposite poses, the first with the front legs raised off the ground, while the second with the back legs up in the air. In both cases, the 3D fixation maps indicate higher fixation values corresponding to the pair of legs raised off the ground, the agent of the represented actions. It appears that the fixation patterns for some characters are clustered on areas related to a potential action which leads to additional questions to explore further.

9 CONCLUSIONS

In this work, we presented a first investigation of the impact of different poses of 3D meshes on gaze data. The aggregated fixation data of the participants obtained by an eye-tracker were projected onto the 3D meshes. Pearson's Correlation Coefficient was used as a measure of similarity between pairs of 3D fixation maps of the same character in different poses and from different views. The obtained 3D fixation maps are coherent with previous studies in that fixation data are focused strongly on the facial regions [How05, McD09, Lav18]. In addition, this paper further indicates that the fixations on the facial regions also appear on characters in different poses. The PCC results show low similarity values between different poses for some of the tested characters. In particular, the low similarity between some of the poses appeared linked with fixations focusing on two factors: geometric salient features and semantically salient parts caused by potential actions or gestures. These results indicate the necessity of further investigations. For example, it would be relevant in future research to explore further the relationship between these two factors and to investigate the influence of different types of actions.

10 ACKNOWLEDGMENTS

This work was supported in part by KK-stiftelsen Sweden, through the ViaTecH Synergy Project (contract 20170056).

11 REFERENCES

[Bor13] Borji, A., and Itti, L. State-of-the-Art in Visual Attention Modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence 35*, 1 (Jan. 2013), 185–207.

[Bro08] Bronstein, A., Bronstein, M., and Kimmel, R. *Numerical Geometry of Non-Rigid Shapes*, 1 ed. Springer Publishing Company, Incorporated, 2008.

[Byl19] Bylinskii, Z., Judd, T., Oliva, A., Torralba, A., and Durand, F. What Do Different Evaluation Metrics Tell Us About Saliency Models? *IEEE Transactions on Pattern Analysis and Machine Intelligence 41*, 3 (March 2019), 740–757.

[Cun11] Cunningham, D. W., and Wallraven, C. *Experimental Design : From User Studies to Psychophysics*. CRC Press LLC, 2011.

[Duc17] Duchowski, A. T. *Eye Tracking Methodology: Theory and Practice*, 3rd ed. Springer Publishing Company, Incorporated, 2017.

[How05] Howlett, S., Hamill, J., and O'Sullivan, C. Predicting and Evaluating Saliency for Simplified Polygonal Models. *ACM Transactions on Applied Perception 2*, 3 (July 2005), 286–308.

[Kim10] Kim, Y., Varshney, A., Jacobs, D. W., and Guimbretière, F. Mesh Saliency and Human Eye Fixations. *ACM Transactions on Applied Perception 7*, 2 (Feb. 2010), 12:1–12:13.

[Kom10] Komogortsev, O. V., Gobert, D. V., Jayarathna, S., Koh, D. H., and Gowda, S. M. Standardization of Automated Analyses of Oculomotor Fixation and Saccadic Behaviors. *IEEE Transactions on Biomedical Engineering 57*, 11 (Nov. 2010), 2635–2645.

[Lar11] Larkin, M., and O'Sullivan, C. Perception of Simplification Artifacts for Animated Characters. In *Proceedings of the ACM SIGGRAPH Symposium on Applied Perception in Graphics and Visualization* (New York, NY, USA, 2011), APGV'11, ACM, pp. 93–100.

[Lav18] Lavoué, G., Cordier, F., Seo, H., and Larabi, M.-C. Visual Attention for Rendered 3D Shapes. *Computer Graphics Forum 37*, 2 (2018), 191–203.

[Lee05] Lee, C. H., Varshney, A., and Jacobs, D. W. Mesh Saliency. In *ACM SIGGRAPH 2005 Papers* (New York, NY, USA, 2005), SIGGRAPH '05, ACM, pp. 659–666.

[Lei16] Leifman, G., Shtrom, E., and Tal, A. Surface Regions of Interest for Viewpoint Selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence 38*, 12 (Dec. 2016), 2544–2556.

[Mes] Cignoni, P., Callieri, M., Corsini, M., Dellepiane, M., Ganovelli, F., and Ranzuglia, G. MeshLab: an Open-Source Mesh Processing Tool. In *Eurographics Italian Chapter Conference* (2008), V. Scarano, R. D. Chiara, and U. Erra, Eds., The Eurographics Association.

[McD09] McDonnell, R., Larkin, M., Hernandez, B., Rudomin, I., and O'Sullivan, C. Eye-catching Crowds: Saliency Based Selective Variation. *ACM Transactions on Graphics 28*, 3 (July 2009), 55:1–55:10.

[Par02] Parkhurst, D., Law, K., and Niebur, E. Modeling the Role of Salience in the Allocation of Overt Visual Attention. *Vision Research 42*, 1 (2002), 107–123.

[Pat16] Patney, A., Salvi, M., Kim, J., Kaplanyan, A., Wyman, C., Benty, N., Luebke, D., and Lefohn, A. Towards Foveated Rendering for Gaze-Tracked Virtual Reality. *ACM Transactions on Graphics 35*, 6 (Nov. 2016), 179:1–179:12.

[Tor06] Torralba, A., Oliva, A., Castelhano, M. S., and Henderson, J. M. Contextual Guidance of Eye Movements and Attention in Real-world Scenes: The Role of Global Features in Object Search. *Psychological Review 113*, 4 (2006), 766–786.

[Tos] TOSCA High-Resolution Dataset. `http://tosca.cs.technion.ac.il/book/resources_data.html`.

[Wan16] Wang, X., Lindlbauer, D., Lessig, C., Maertens, M., and Alexa, M. Measuring the Visual Salience of 3D Printed Objects. *IEEE Computer Graphics and Applications 36*, 4 (July 2016), 46–55.

[Wan18] Wang, X., Koch, S., Holmqvist, K., and Alexa, M. Tracking the Gaze on Objects in 3D: How Do People Really Look at the Bunny? *ACM Transactions on Graphics 37*, 6 (Dec. 2018), 188:1–188:18.

[Wu13] Wu, J., Shen, X., Zhu, W., and Liu, L. Mesh Saliency with Global Rarity. *Graphical Models 75*, 5 (2013), 255–264.

Custom Dilated Edge Detection Filters

Victor BOGDAN
West University of
Timişoara
Bd. V. Pârvan 4
Timişoara, RO-300223
Romania.
victor.bogdan97@e-uvt.ro

Cosmin BONCHIŞ
West University of
Timişoara and
The eAustria Research
Institute,
Bd. V. Pârvan 4, 045B,
Timişoara, RO-300223
Romania.
cosmin.bonchis@e-uvt.ro

Ciprian ORHEI
Politehnica University of
Timisoara,
Timişoara, RO-300223
Romania.
ciprian.orhei@cm.upt.ro

ABSTRACT

Edge detection is a basic and fundamental function in image processing. Inspired by the new dilated convolution techniques which have impressive results in machine learning, we discuss here the idea of dilating the standard filters which are used to compute the gradient of an image. We compare the result of the proposed dilated filters with original filters and extend or custom variants of them. We also present the impact of dilatation on a complex edge detection algorithm, like Canny.

Keywords
Dilated filters, edge detection operator, edge detection, Canny algorithm.

1 INTRODUCTION

The edge of an image is the most basic feature on the image and has been intensively researched. A variety of mathematical methods have been used to identify points in which the image brightness changes sharply or has discontinuities. This is a fundamental tool in image processing, image analysis, machine vision and computer vision, particularly in the areas of feature detection and feature extraction.

Standard edge detection filters are built for highlighting intensity change boundaries in the near neighborhood image regions. The most frequently used methods we can find in: [Sobel and Feldman, 1973, Roberts, 1963, Prewitt, 1970, Scharr, 2000]. Those filters are used in many computer vision algorithms which rely on edge detection for application as face recognition, target recognition, obstacle detection, image compression etc.

The extended version of the standard filters can be find in many research article in the literature. [Gupta and Mazumdar, 2013, Aybar et al., 2006] contains the mathematical proof of how to extend the Sobel operator and the performance of that extension. In [Lateef, 2008] we can find the definition of extended Sobel and Prewitt operators to a 5×5 mask in order to obtain more continuous edges. In [Levkine, 2012] we are presented with the analytical extension of Prewitt, Sobel and Scharr operators. [Kekre and Gharge, 2010] uses the extension of Prewitt, Kirsch and Sobel operators to increase the tumor area in mammography images. In all these papers, we found that a bigger filter region is useful in order to find more accurate edges.

Merging those extension ideas with the recently dilated convolution techniques in machine learning, we propose not to extend but to simply **dilate** standard edge detection filters mentioned above. The dilated convolution methods have been proven very beneficial in many highly cited computer-vision papers: for small objects detection [Hamaguchi et al., 2018], in dense prediction tasks [Chen et al., 2018, Chen et al., 2015], on prediction without losing resolution [Yu and Koltun, 2016], for feature classifications in time series [Yazdanbakhsh and Dick, 2019] or beneficial for context aggregation [Zhao et al., 2017].

In this paper,we define an expansion of filters, by simply adding zeros in the expansion gapes and we call it dilation.This method of dilation is neither from the well known mathematical morphological sense, from [Haralick et al., 1987], nor the geometric extension of the kernels discussed above.

To evaluate the dilation benefits to an edge detection filter,we restrict our analysis only to the first order derivative gradient-based edge detection filters. The details regarding the extended versions of some edge detection filters selected by us can be found in section 2. In section 3, we present the defined dilated filters that we will use in this paper. Sections 4 and 5 highlight the results of our hypothesis regarding dilating of the filters, rather than extending them.

2 PRELIMINARIES

In this paper, we will present the dilated filters Prewitt [Prewitt, 1970], Scharr [Scharr, 2000]

https://doi.org/10.24132/CSRN.2020.3001.19

and Sobel [Sobel and Feldman, 1973], used to compute the gradient of an image in order to find its edges. We compare the result of our dilated filters with the standard and extended filters presented in [Gupta and Mazumdar, 2013, Lateef, 2008, Levkine, 2012, Kekre and Gharge, 2010, Aybar et al., 2006], (presented in Figures 1 to 6), using different image sets. To determine the impact of the dilated filters, we modify the Canny Edge detection algorithm [Canny, 1986] in order to use the dilated or extended version of each filter for calculating gradients. For a better comparison of the results, we used the BSDS500 benchmark tool and image sets from [Arbelaez et al., 2011].

Gradient operators

We consider the standard filters presented in Figures 1, 2, 3, where Gx and Gy are the gradients masks. In [Lateef, 2008, Kekre and Gharge, 2010, Levkine, 2012, Aybar et al., 2006] we can find the extended filter we will use to compare, Figures 4 to 6.

$$\begin{bmatrix} 3 & 0 & -3 \\ 10 & 0 & -10 \\ 3 & 0 & -3 \end{bmatrix} \begin{bmatrix} 3 & 10 & 3 \\ 0 & 0 & 0 \\ -3 & -10 & -3 \end{bmatrix}$$

Figure 1: Scharr Gx and Gy kernels

$$\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

Figure 2: Prewitt Gx and Gy kernels

$$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Figure 3: Sobel Gx and Gy kernels

We will consider the following standard formula where the Gx and Gy gradient components are used to define the gradient magnitude $|G|$ in Equation 1.

$$|G| = \sqrt{Gx^2 + Gy^2}. \tag{1}$$

We use the edge detection algorithm steps, presented in [Woods, 2011], which are generally used to convolve filters with a source image in order to obtain the edge image:

$$\begin{bmatrix} 1 & 1 & 0 & -1 & -1 \\ 2 & 2 & 0 & -2 & -2 \\ 3 & 6 & 0 & -6 & -3 \\ 2 & 2 & 0 & -2 & -2 \\ 1 & 1 & 0 & -1 & -1 \end{bmatrix} \begin{bmatrix} 1 & 2 & 3 & 2 & 1 \\ 1 & 2 & 6 & 2 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ -1 & -2 & -6 & -2 & -1 \\ -1 & -2 & -3 & -2 & -1 \end{bmatrix}$$

Figure 4: 5×5 Scharr Gx and Gy kernels

$$\begin{bmatrix} 2 & 1 & 0 & -1 & -2 \\ 2 & 4 & 0 & -4 & -2 \\ 2 & 4 & 0 & -4 & -2 \\ 2 & 4 & 0 & -4 & -2 \\ 2 & 1 & 0 & -1 & -2 \end{bmatrix} \begin{bmatrix} 2 & 2 & 2 & 2 & 2 \\ 1 & 4 & 4 & 4 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & -1 & -1 & -1 \\ -2 & -4 & -4 & -4 & -2 \end{bmatrix}$$

Figure 5: 5×5 Prewitt Gx and Gy kernels

$$\begin{bmatrix} 5 & 4 & 0 & -4 & -5 \\ 8 & 10 & 0 & -8 & -10 \\ 10 & 20 & 0 & -20 & -10 \\ 8 & 10 & 0 & -8 & -10 \\ 5 & 4 & 0 & -4 & -5 \end{bmatrix} \begin{bmatrix} 5 & 8 & 10 & 8 & 5 \\ 4 & 10 & 20 & 10 & 4 \\ 0 & 0 & 0 & 0 & 0 \\ -4 & 10 & -20 & -8 & -4 \\ -5 & -8 & -10 & -8 & -5 \end{bmatrix}$$

Figure 6: 5×5 Sobel Gx and Gy kernels

Step 1 Convert the image to gray-scale, preparing it as an input for Sobel filter convolution.

Step 2 Reduce the noise in the source image by applying the Gaussian filter, in order to obtain smooth continuous values.

Step 3 Applying the filters by convolving the gray-scale image with their kernels on the x and y axes and then applying the gradient magnitude (Equation 1).

Step 4 Each pixel, which has an intensity value higher or equal to a *threshold*, will have its value set to MaxValue (e.g. 255), else to 0, therefore the edges will be represented by the white pixels.

Canny edge operator

Furthermore, the Canny edge detection algorithm [Canny, 1986] is a widely known and one of the most used edge detection algorithms.

Step 1 Convert the image to gray-scale

Step 2 Applying the Gaussian Filter, in order to obtain smooth continuous values.

Step 3 Applying the filters by convolving the grayscale image with their kernels on the *x* and *y* axes.

Step 4 Non-maximum suppression, for edge thinning of the obtained results.

Step 5 Edge tracking by hysteresis using double threshold.

The double threshold is found by using the maximum pixel intensity in the input image and applying the formula from Equation 2, similar with the equation defined in [Xu et al., 2011]. T_h is the upper threshold, T_l is the lower threshold and $max(input)$ is the maximum pixel intensity in the input image. From our experiments the best results were obtained with the fixed weights values $w_h = 0.7$ and $w_l = 0.3$.

$$\begin{aligned} T_h &= max(input) \times w_h \qquad (2)\\ T_l &= T_h \times w_l \end{aligned}$$

Benchmarking the edge operators

For highlighting the results obtained,we use BSDS500 [Arbelaez et al., 2011] which contains a dataset of natural images that have been manually segmented. The human annotations serve as ground truth for the benchmark for comparing different segmentation and boundary detection algorithms. For evaluating the images generated from algorithms to the ground truth images the Corresponding Pixel Metric (CPM) algorithm [Prieto and Allen, 2003] is used. This metric is reliable for correlating similarities with a small localization error in the detected edges. The metric first finds an optimal matching of the pixels between the edge images and then estimate the error produced by this matching.

The benchmark [Arbelaez et al., 2011] uses 500 test images, which are split in 3 different sets, each having at least 5 human segmented boundary ground-truth images.

For each image two quantities Precision (P) and Recall (R) will be computed, as were defined in [Sasaki, 2007]. Precision, with formula 3, represents the probability that a resulting edge/boundary pixel is a true edge/boundary pixel. Recall, with formula 4, represents the probability that a true edge/boundary pixel is detected. Where TP (True Positive) represents the number of matched edge pixel, FP (False Positive) the number of edge pixels which are incorrectly highlighted and FN (False Negative) the number of pixel that have not been detected.

$$P = \frac{TP}{TP+FP}. \qquad (3)$$

$$R = \frac{TP}{TP+FN}. \qquad (4)$$

Those two quantities are used to compute *F-measure* (F1-score) by applying the formula 5, as is defined in [Arbelaez et al., 2011].

$$F-measure = \frac{2*TP}{2*TP+FP+FN}. \qquad (5)$$

3 DILATED FILTERS

One of the commonly used methods in detecting the edges in images is by convolving the initial image with an edge detection operator. The filter highlights the difference between the pixel intensities. To obtain better results, we can combine kernels which will result in a higher change in pixel intensity.

Definition 1. *A dilated filter is obtained by expanding the original filter by a dilation factor/size.*

In this paper, we propose to use dilated filters for detecting the edges in images that are obtained by the Definition 1. The standard 3 × 3 kernels from Sobel, Prewitt and Scharr are dilated to 5 × 5, 7 × 7 filters, for both axes. By dilating the kernels, we propose to increase the distance between the pixels,distance which influences the result of the convolution. This expansion induces the possibility of finding stronger intensity changes in the image. When we dilate the kernels, we are filling the newly added positions with 0s, as one can see the Figures 7 to 12.

$$\begin{bmatrix} 1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & -2 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 2 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & -2 & 0 & -1 \end{bmatrix}$$

Figure 7: Sobel 5 × 5 dilated Gx and Gy filterss

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & -2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 2 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & -2 & 0 & 0 & -1 \end{bmatrix}$$

Figure 8: Sobel 7 × 7 dilated Gx and Gy filters

With our approach,in most of the test cases,we obtain better edge detection results than the filters presented in

$$\begin{bmatrix} 3 & 0 & 0 & 0 & -3 \\ 0 & 0 & 0 & 0 & 0 \\ 10 & 0 & 0 & 0 & -10 \\ 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & -3 \end{bmatrix} \begin{bmatrix} 3 & 0 & 10 & 0 & 3 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -3 & 0 & -10 & 0 & -3 \end{bmatrix}$$

Figure 9: Scharr 5 × 5 dilated Gx and Gy filters

[Gupta and Mazumdar, 2013] or [Levkine, 2012]. Another advantage that we noticed was the fact that even if the filter size has increased, the number of calculation hasn't, fact that resulted in low run time compared to extended 5 × 5 or 7 × 7 filters. Even if the proposed filters don't respect the geometrical gradient formulas and the rules of extension, the good results in edge finding suggested that dilating is a solution to take in consideration, rather than extending.

$$\begin{bmatrix} 3 & 0 & 0 & 0 & 0 & 0 & -3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 10 & 0 & 0 & 0 & 0 & 0 & -10 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 & 0 & -3 \end{bmatrix} \begin{bmatrix} 3 & 0 & 0 & 10 & 0 & 0 & 3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -3 & 0 & 0 & -10 & 0 & 0 & -3 \end{bmatrix}$$

Figure 10: Scharr 7 × 7 dilated Gx and Gy filters

$$\begin{bmatrix} 1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & -1 & 0 & -1 \end{bmatrix}$$

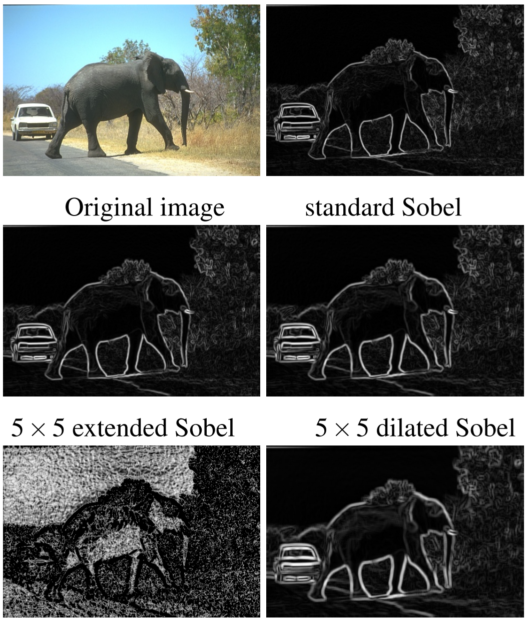
Figure 11: Prewitt 5 × 5 dilated Gx and Gy filters

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & -1 & 0 & 0 & -1 \end{bmatrix}$$

Figure 12: Prewitt 7 × 7 dilated Gx and Gy filters

4 FILTER CONVOLUTION RESULTS

The following section consists of comparisons between the results of convolving an image with the standard, extended and dilated filters presented until now. We present first some visual comparison results of the dilated filters, in Figures 13, 14, 15. By using the benchmarking tool from BSDS500 we show the statistical results by comparing the test images with ground truth, as presented in [Arbelaez et al., 2011].

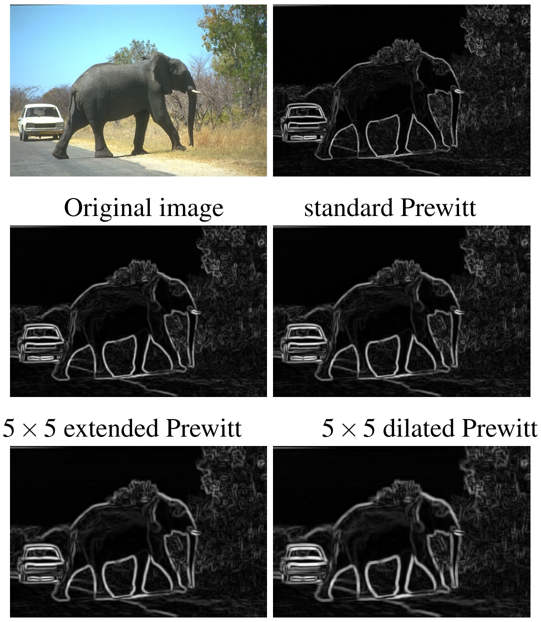


Original image standard Sobel

5 × 5 extended Sobel 5 × 5 dilated Sobel

7 × 7 extended Sobel 7 × 7 dilated Sobel

Figure 13: Standard, Extended and Dilated Sobel Results

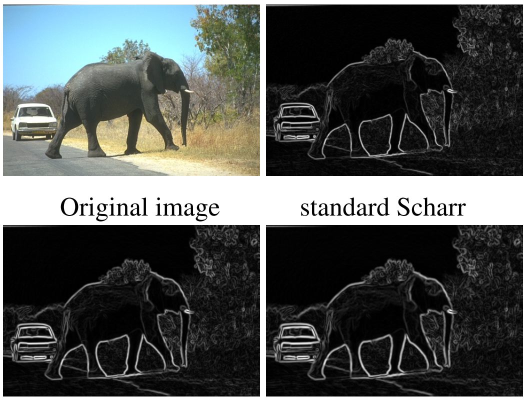


Original image standard Prewitt

5 × 5 extended Prewitt 5 × 5 dilated Prewitt

7 × 7 extended Prewitt 7 × 7 dilated Prewitt

Figure 14: Standard, Extended and Dilated Prewitt Results



Original image standard Scharr

5 × 5 extended Scharr 5 × 5 dilated Scharr

Figure 15: Standard, Extended and Dilated Scharr Results

First results

From Figures 13 to 15 we can observe that when we convolve the original source image with the dilated filters, we obtain more pixels with a high gradient magnitude than the extended ones. Also, the more the dilation factor is increased, the more the edge images seem to lose the details while the edges seem to get blurry. Visually, there seems to be a compromise between details and the newly detected edges, see other detailed examples in Appendix in Figure 21.

We can observe some differences between the standard or extended filters and our dilated filters, however for a rigorous comparison we use the BSDS500 benchmark [Arbelaez et al., 2011], in order to observe the impact of the dilatation on a data base of ground truth images.

Benchmark results

Our dilated filter results have been compared to the ground-truth images. The benchmark chooses a number of threshold values between 0 and 255 for creating a binary map of the edge algorithm output. The best overall F1-score is chosen from the variants of the output for each threshold. That gives us the rule for choosing the best possible result for each filter for each image in the set. For each threshold the two quantities Precision and Recall will be computed.

Tables 1 to 3 contain the results of the standard, extended and dilated filters on the BSDS500 train set. The overall recall, precision and F1-score represents the average for all samples.

We highlight that every **dilated filter**, with one exception the Prewitt extended 7 × 7, **obtained a better overall F1-score** than the extended filters. This can be explained by the sparsity of the dilated filter that induces more edges and less noise.

Another advantage of using the dilated filters instead of the standard or extended ones is given by runtime simulation results.

Filter Size	Overall Recall	Overall Precision	Overall F1-score
Standard 3 × 3	0.48723	0.54077	0.51261
Extended 5 × 5	0.79035	0.42298	0.55105
Dilated 5 × 5	0.78942	0.43077	**0.55738**
Extended 7 × 7	0.77057	0.46317	**0.57857**
Dilated 7 × 7	0.79008	0.44572	0.56992

Table 1: Prewitt Comparison on the Test Set

Filter Size	Overall Recall	Overall Precision	Overall F1-score
Standard 3 × 3	0.48714	0.53631	0.51054
Extended 5 × 5	0.79783	0.40453	0.53685
Dilated 5 × 5	0.78892	0.42243	**0.55023**

Table 2: Scharr Comparison on the Test Set

Filter Size	Overall Recall	Overall Precision	Overall F1-score
Standard 3 × 3	0.48863	0.53765	0.51197
Extended 5 × 5	0.79724	0.40666	0.53860
Dilated 5 × 5	0.79021	0.42658	**0.55407**
Extended 7 × 7	0.85249	0.17964	0.29675
Dilated 7 × 7	0.78627	0.44633	**0.56942**

Table 3: Sobel Comparison on the Test Set

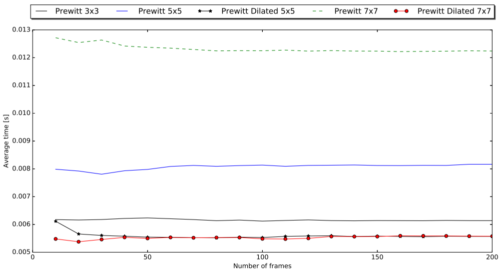


Figure 16: Prewitt average runtime

Runtime results

We applied all the filters presented above by using our framework using Python and the results are presented in the graphics from Figures 16, 17, 18.

From the plots one can see that the runtime doesn't increase proportional with the size of the dilated filter. This stability can be explained by the fact that dilation

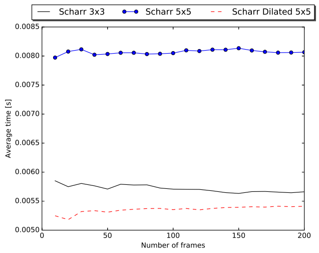


Figure 17: Scharr average runtime

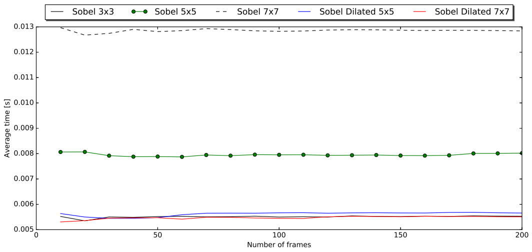


Figure 18: Sobel average runtime

doesn't increase the number of necessary operations to be executed.

In this section,we set forth that dilating the filters is a good compromise for detecting edges instead of extending them, discovering benefits regarding the edge versus noise output and runtime results.

5 CANNY SIMULATION RESULTS

In this section, we use the Canny edge detection algorithm to compare the results of convolving an image with the standard, extended and dilated Sobel filters. Similar with the previous analysis, we present a visual comparison, in Figure 19 and the statistic results using BSDS500 [Arbelaez et al., 2011], see the Tables 4 and 5.

First results

From Figure 19, we can observe that applying the dilated Sobel filter in a more complex edge detection algorithm like Canny [Canny, 1986] produces visible enhancements of its results. We can also observe that by using the dilated filters the results contain less noise than the extended ones.

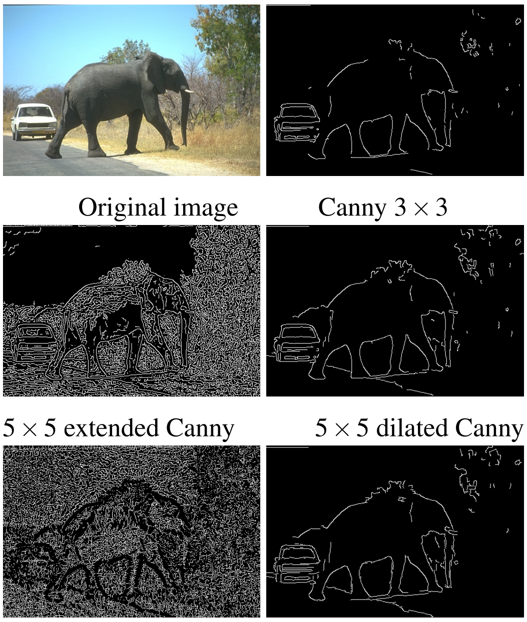
Benchmark results

In Table 4, we show the results of the Canny edge detection algorithm using the standard, extended and dilated Sobel filters. We can notice that the custom dilated filters have better F1-scores than the standard or extended Sobel filters when they are used together with the Canny edge detection algorithm.

The results illustrated in Table 4 show that, by increasing the dilated factor, we can obtain a better F1-score, see the 7 × 7 dilated results.

Filter Size	Overall Recall	Overall Precision	Overall F1-score
Standard 3×3	0.47400	0.54822	0.50842
Extended 5×5	0.99408	0.20711	0.34280
Dilated 5×5	0.59956	0.55094	**0.57422**
Extended 7×7	0.89121	0.17991	0.29938
Dilated 7×7	0.60456	0.56940	**0.58645**

Table 4: Canny using Sobel comparison on the Test Set



Original image

Canny 3 × 3

5 × 5 extended Canny

5 × 5 dilated Canny

7 × 7 extended Canny

7 × 7 dilated Canny

Figure 19: Standard, Extended and Dilated Canny Results

Filter Size	Overall Recall	Overall Precision	Overall F1-score
Standard 3×3	0.47400	0.54822	0.50842
Dilated 5×5	0.59956	0.55094	0.57422
Dilated 7×7	**0.60456**	**0.56940**	**0.58645**
Dilated 9×9	0.6010	0.57216	0.58622
Dilated 11×11	0.6011	0.56266	0.58125
Dilated 13×13	0.5964	0.54197	0.56787
Dilated 15×15	0.5901	0.51935	0.55247

Table 5: Canny using Sobel dilated filters on the Test Set

By continuing to dilate the Sobel filter, we can also notice that the F1-score starts to decrease from a certain point. Therefore, in Table 5 one can see that 9 × 9, 11 × 11, 13 × 13 and 15 × 15 dilated Sobel filters obtained a significantly lower F1-score than the 7 × 7 dilated filter. This is also the case for the overall precision, whereas the overall recall alternates but there are no significant differences.

Run time results

From the simulation results, illustrated in Figure 20, we observe that neither in Canny simulation the runtime was not impacted by the size of the dilated filter.

The Canny approach validates our hypothesis that dilating the filters is better than extending them. We achieve here the good results which we expected from dilated filters regarding edge discovering and runtime.

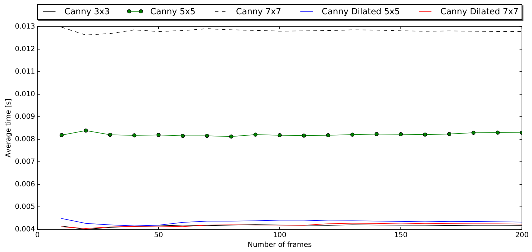


Figure 20: Canny average runtime

6 CONCLUSIONS AND FUTURE WORK

In both comparisons, statistical and visual, we can observe that by dilating the filters, rather than extending them, helps to find more edge pixels than the standard filters. By dilating the kernels of the Sobel or Scharr filter, we obtained a better recall and precision, which can be observed in Tables 2 and 3, and thus a better F1-score.

The biggest improvement of the dilated filters can be seen in the comparison between the extended and dilated versions of the Sobel filter in the Canny algorithm, see Table 4.

From the evolution of the metrics in Table 5 we can observe that we benefit from dilating the filters up to a certain point. Someone can say that 9×9 is better than 7×7 if we look at the Precision and F1-score. Therefore, we would say that the best dilation depends on the input images and the nature of application in which we desire edge detection.

By visually and statistically comparing the dilated filters with the standard and extended filters, we can conclude that the dilated filters can achieve better results, in most cases.

Because of the simple structure of the custom dilated filters, they are also a good choice when the runtime matters. The other filters from [Gupta and Mazumdar, 2013, Lateef, 2008, Levkine, 2012, Kekre and Gharge, 2010, Aybar et al., 2006] require a larger number of operations in order to return the resulting edge pixels, whereas the custom dilated filters have always the same number of operations for any extension.

From the experiments that have been done we saw that dilated filters might not be very efficient in images which contain many details because the dilation of the kernels can cause loss of details in images. Those images that have high differences of pixel intensities in a small range in different regions are the hard scenario for dilated filter and could expect more investigation.

As an open topic for a further investigation,it could consist in defining the similar dilated filters on second order derivative edge detection operators or more complex filters, with focus on the performance evaluation.

7 ACKNOWLEDGEMENTS

The authors would like to thank Oana Rotaru-Orhei for her comments and the three anonymous reviewers for their insightful suggestions. This work was partially supported by a grant of Ministry of Research and Innovation, CNCS - UEFISCDI, project number PN-III-P4-ID-PCE-2016-0842, within PNCDI III.

8 REFERENCES

[Arbelaez et al., 2011] Arbelaez, P., Maire, M., Fowlkes, C., and Malik, J. (2011). Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):898–916.

[Aybar et al., 2006] Aybar, E. et al. (2006). Sobel edge detection method for matlab. *Anadolu University, Porsuk Vocational School*, 26410.

[Canny, 1986] Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698.

[Chen et al., 2015] Chen, L., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2015). Semantic image segmentation with deep convolutional nets and fully connected crfs. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

[Chen et al., 2018] Chen, L., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2018). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(4):834–848.

[Gupta and Mazumdar, 2013] Gupta, S. and Mazumdar, S. G. (2013). Sobel edge detection algorithm. *International journal of computer science and management Research*, 2(2):1578–1583.

[Hamaguchi et al., 2018] Hamaguchi, R., Fujita, A., Nemoto, K., Imaizumi, T., and Hikosaka, S. (2018). Effective use of dilated convolutions for segmenting small object instances in remote sensing imagery. In *2018 IEEE Winter Conference on Applications of Computer Vision, WACV 2018, Lake Tahoe, NV, USA, March 12-15, 2018*, pages 1442–1450. IEEE Computer Society.

[Haralick et al., 1987] Haralick, R. M., Shapiro, L., and Lee, J. (1987). Morphological edge detection. *IEEE J. Robotics Automat*, 3:142–155.

[Kekre and Gharge, 2010] Kekre, H. and Gharge, S. (2010). Image segmentation using extended edge operator for mammographic images. *International journal on computer science and Engineering*, 2(4):1086–1091.

[Lateef, 2008] Lateef, R. A. R. (2008). Expansion and implementation of a 3x3 sobel and prewitt edge detection filter to a 5x5 dimension filter. *Journal of Baghdad College of Economic sciences University*, (18):336–348.

[Levkine, 2012] Levkine, G. (2012). Prewitt, sobel and scharr gradient 5x5 convolution matrices. *Image Process. Articles*.

[Prewitt, 1970] Prewitt, J. M. (1970). Object enhancement and extraction. *Picture processing and Psychopictorics*, 10(1):15–19.

[Prieto and Allen, 2003] Prieto, M. and Allen, A. (2003). A similarity metric for edge images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25:1265– 1273.

[Roberts, 1963] Roberts, L. G. (1963). *Machine perception of three-dimensional solids*. PhD thesis, Massachusetts Institute of Technology.

[Sasaki, 2007] Sasaki, Y. (2007). The truth of the f-measure. Technical report, School of Computer Science, University of Manchester.

[Scharr, 2000] Scharr, H. (2000). *Optimal operators in digital image processing*. PhD thesis, Ruprecht-Karls-Universitat Heidelberg, Germany.

[Sobel and Feldman, 1973] Sobel, I. and Feldman, G. (1973). *Pattern Classification and Scene Analysis*, pages 271–272.

[Woods, 2011] Woods, J. W. (2011). *Multidimensional Signal, Image, and Video Processing and Coding, Second Edition*. Academic Press, Inc., Orlando, FL, USA, 2nd edition.

[Xu et al., 2011] Xu, Q., Chakrabarti, C., and Karam, L. J. (2011). A distributed canny edge detector and its implementation on fpga. In *2011 Digital Signal Processing and Signal Processing Education Meeting (DSP/SPE)*, pages 500–505.

[Yazdanbakhsh and Dick, 2019] Yazdanbakhsh, O. and Dick, S. (2019). Multivariate time series classification using dilated convolutional neural network. *CoRR*, abs/1905.01697.

[Yu and Koltun, 2016] Yu, F. and Koltun, V. (2016). Multi-scale context aggregation by dilated convolutions. In Bengio, Y. and LeCun, Y., editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

[Zhao et al., 2017] Zhao, H., Shi, J., Qi, X., Wang, X., and Jia, J. (2017). Pyramid scene parsing network. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 6230–6239. IEEE Computer Society.

APPENDIX

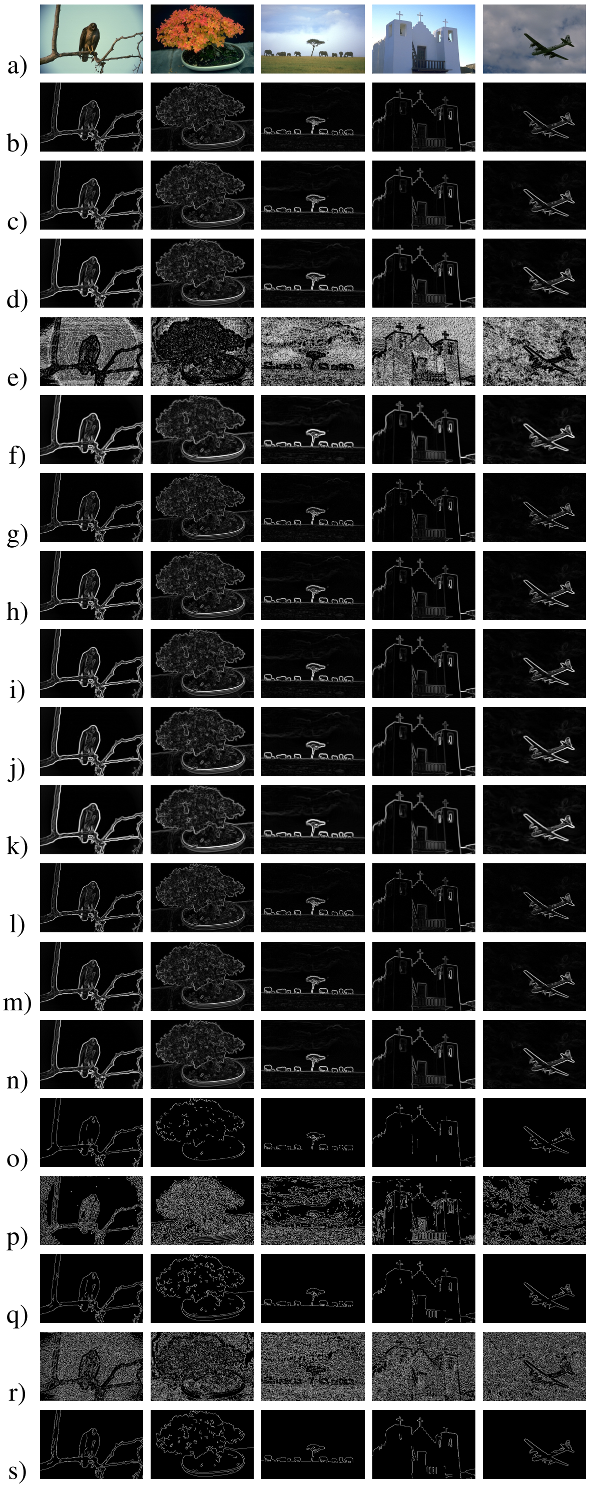


Figure 21: a) Original image, b) 3 × 3 Sobel, c) 5 × 5 extended Sobel, d) 5 × 5 dilated Sobel, e) 7 × 7 extended Sobel, f) 7 × 7 dilated Sobel, g) 3 × 3 Prewitt, h) 5 × 5 extended Prewitt, i) 5 × 5 dilated Prewitt, j) 7 × 7 extended Prewitt, k) 7 × 7 dilated Prewitt, l) 3 × 3 Scharr, m) 5 × 5 extended Scharr, n) 5 × 5 dilated Scharr, o) 3 × 3 Canny, p) 5 × 5 extended Canny, q) 5 × 5 dilated Canny, r) 7 × 7 extended Canny, s) 7 × 7 dilated Canny.

Covariance Based Differential Geometry Segmentation Techniques for Surface Representation Using Vector Field Framework

Mana Eskandari[1] Denis Laurendeau[1]
[1]Computer Vision and System Laboratory, Dept. of ECE, Laval University
1065 Av. De la Médicine
Québec City, Canada, G1V 0A6
mana.eskandari.1@ulaval.ca
denis.lauredeau@gel.ulaval.ca

ABSTRACT

In this paper, the concepts of differential geometry traditionally applied to the segmentation of range maps is revisited in the context of implicit surface representation of unorganized point clouds. The paper shows that it is possible to combine covariance-based differential geometry and implicit surface representation methods to perform the segmentation of an unorganized point cloud (and not just a range map) into seven surface types. The acquisition of the point cloud data is achieved with handheld scanners used in metrology applications. The advantages of combining covariance-based differential geometry and implicit surface representation are that the segmentation does not require surface fitting nor does it require that all points be processed, thus reducing computational complexity. The segmentation approach is validated on synthetic data as well as point clouds borrowed from common datasets. Scans obtained from commercial metrologic handheld 3D sensors are also used for validation. The paper first presents the workflow commonly used for 3D scanning using handheld 3D scanners in the context of metrology. This is followed by a discussion on the different methods that are used for surface representation including the vector field, the implicit representation method exploited in this paper. Basic concepts of classical differential geometry for surface segmentation are presented. This is followed by the presentation of covariance-based differential geometry. The concepts of handheld 3D scanning, covariance-based differential geometry and implicit surface representation are then combined to achieve efficient segmentation of a point cloud into seven different surface types. Experimental results obtained on synthetic 3D data as well as real data demonstrate the segmentation approach.

Keywords

3D reconstruction, volumetric representation, geometry modeling.

1. INTRODUCTION

Applied metrology consists in the application of measurements in different fields such as quality control, inspection, product design and reverse engineering. If accuracy and precision are two important components of metrology, the time needed to achieve the measurements is a relevant issue, especially in the context of quality control in an industrial context for which a large number of parts have to be processed. The ease with which the scan can be obtained by users is also important when such users are domain specialists but not necessarily experts in 3D scanning.

Over the years, 3D sensors have become very popular because they are cheaper and easier to use than classical Coordinate Measuring Machines (CMM) while still achieving metrologic accuracy. In addition, 3D sensors can capture dense point clouds that convey the geometry of the object in real-time. Comparatively, capturing dense 3D point clouds with a CMM can be a very tedious and time consuming process. Capturing the geometry of parts is not only important in metrology but is also useful in many fields such as reverse engineering, design intent assessment and graphics rendering.

Among 3D sensors, handheld 3D sensors are of great interest because they allow the capture of 3D data on the specimens to be inspected in a very natural way which, in many aspects, resembles spray painting. As shown in Fig.1, scanning an object with a handheld 3D sensor consists in moving the sensor around the surface of the object of interest while the 3D coordinates of points at the surface are collected.

Most 3D handheld sensors are active sensors that project some sort of light pattern (laser point, laser stripes, laser crosses, white light patterns, Moiré fringes, etc., see [DANE2018] for an overview of different 3D sensing technologies) on the object to ease the image analysis process leading to the measurement of 3D coordinates. The 3D coordinates of points acquired from a pose (i.e. position and orientation) of the sensor are expressed in this local

https://doi.org/10.24132/CSRN.2020.3001.20

reference frame. The estimation of the rigid transformation (rotation and translation) between each pose of the sensor and a "global" reference frame is needed if the 3D points are to be expressed in a common reference frame. This global reference frame can be the initial pose of the sensor when the scanning process starts. The estimation of the rigid transformations is made easier if a real-time self-positioning strategy is used to compute the position and orientation of the sensor with respect to the object. One way of implementing self-positioning is to install markers (often retroreflective markers) at the surface of the object and to estimate the pose of the sensor with respect to these markers. The rigid transformation between different poses of the sensor is then readily available by exploiting registration algorithms such as the well-known Iterative Closest Point (ICP) approach [RUSI2001].

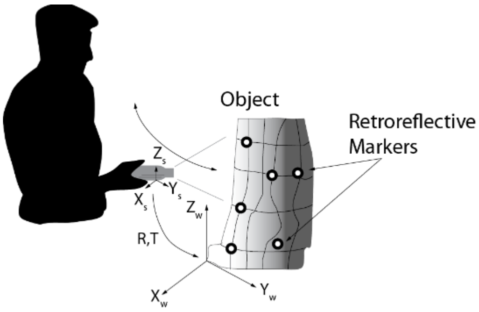


Figure 1. The 3D scanning process using a handheld scanner

Surface Representation

Once the cloud of unorganized 3D points covering the entire surface of the object has been captured, a model of the surface must be built if metrologic measurements are to be performed on the object. This model can also be used to analyze the geometry of the object. Two main representations can be exploited to build this model: *explicit* representations and *implicit* representations. The former representation makes the geometry supported by the point cloud explicit, for instance by building a triangular mesh [CHEN2012]. Such a mesh contains the connectivity between points in the cloud and is a compact representation of the geometry of the surface. This connectivity can take the form of a vertex-triangle list and a triangle-vertex list and can also include the information on the normal to the surface at each vertex. The latter representation rather encodes the geometric information contained in the point cloud implicitly into a volumetric structure composed of voxels. Two main types of voxel-based implicit representations have been proposed: the distance field [CURL2996] and the vector field [TUBI2002]. The volumetric structure must be processed a posteriori to produce a mesh representing the geometry explicitly. The Marching Cubes algorithm is often used for this task [LORE1987].

When handheld 3D sensors are used for *real-time* modelling (i.e. the model of the surface is built as the 3D points are measured), three tasks must be achieved: *i)* view registration, *ii)* view integration and *iii)* model visualization. View registration consists in the estimation of the rigid transformation between points of view from which the 3D data is collected. View integration aims at merging redundant 3D data common to two or more views. Finally, model visualization is the task of rendering the 3D model as it is being built so the user can observe the progression of the scan and plan the scanning strategy as points are being collected.

The advantage offered by an *explicit* representation is that a low-level model is readily available. However, it is not adapted to real-time modelling. The main reason for this is that the registration and integration steps rely on finding nearest neighbours and that the search for nearest neighbours to a point becomes too computationally expensive when the number of points increases. Updating a mesh as new 3D points are being collected is also impossible to achieve in real time. The advantage of using *implicit* representations is that some, such as the vector field, have demonstrated the ability to support the three modelling steps in real-time that cannot be achieved by other methods [KHAK2019]. As described next, a major advantage of the vector field representation is that it encodes the surface normal as well as information on the nearest neighbors in each voxel, thus enabling nearest neighbor search in linear time complexity. However, if an accurate model needs to be built, the voxel size must be small which may lead to huge memory requirements.

The rest of the paper is organized as follows. Related work is detailed in Section 2. The proposed method is explained in Section 3. Section 4 presents the experimental results which demonstrate the performance of the method. Section 5 concludes and proposes future work.

2. RELATED WORK

A Review of the Vector Field Implicit Representation for Real-time Modelling [TUBIC2002]

As shown in Fig.2, the vector field is composed of a regular grid made of cubic voxels with side length *L*.

Each voxel in the grid is addressed by the coordinates of its center v_{ijk} in a reference frame W_r. Let us assume that 3D points with coordinates $\mathrm{p_m}$ on the surface *S* are collected by the sensor in frame W_r. The covariance matrix $C_{i,j,k}$ of the points falling in voxel *ijk* is defined as in Eq. (1):

$$C_{i,j,k} = \frac{1}{N}\sum_{u=1}^{N}\left(p_u - \overline{p}\right)\left(p_u - \overline{p}\right)^t \qquad (1)$$

where $\overline{p}$ is the mean vector as defined in Eq. (2).

$$\overline{p} = \frac{1}{N}\sum_{u=1}^{N} p_u \qquad (2)$$

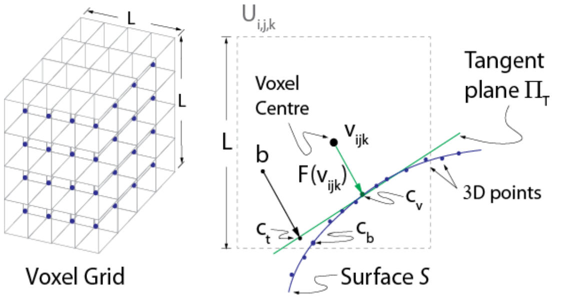


Figure 2. The Vector Field implicit representation (Adapted from [TUBI2002]))

If the voxel size *L* is small enough, it is assumed that the object surface in the voxel can be approximated by the plane Π_T tangent to the surface. The normal vector to tangent plane Π_T is the eigenvector corresponding to the smallest eigenvalue λ_{min} of $C_{i,j,k}$. $F(V_{i,j,k})$, c_v and $C_{i,j,k}$ are stored in each voxel. In the voxel, point c_v on the tangent plane that is closest to $v_{i,j,k}$ is given by Eq. (3).

$$c_v = F(V_{i,j,k}) + V_{\mathrm{i,j,k}} \qquad (3)$$

As new points are collected by the sensor, $C_{i,j,k}$, λ_{min}, $F(V_{i,j,k})$ and c_v can be updated in real-time. Now, let us assume that the closest point c_t to the surface approximated by Π_T has to be found for a point *b* falling in voxel *i,j,k*. The coordinates of c_t can be computed from the content of the vector field with Eq. (4) where $\langle , \rangle$ represents the scalar product and $\| \ \|$ is the norm of a vector.

$$c_t = b + F(V_{i,j,k}) + \frac{\langle F(V_{i,j,k}), V_{\mathrm{i,j,k}} - b\rangle}{\|F(V_{\mathrm{i,j,k}})\|} \qquad (4)$$

As shown in Fig.2, the closest point c_t estimated by Eq. (4) is a very good approximation of the true closest point on the surface c_b. In addition, for a single unit of data *b*, the computational complexity of finding its closest point on the surface is constant *O(1)* and is of order *O(n)* for *n* units of data (i.e. *n* points for which the closest points on the surface needs to be computed). This is more efficient than classical nearest neighbor finding approaches which show $O(n^2)$ or *O(n log(n))* computational complexity. The vector field representation thus allows the view registration and view integration steps to execute in real-time. As mentioned above, with the vector field representation, the price to pay for computational efficiency is the amount of memory that is needed to store the voxel grid at a resolution for which the planar approximation is valid.

High-Level Surface Segmentation Using Differential Geometry

An explicit representation such as a triangular mesh is a low-level model of a surface that provides the connectivity between points and that is good for visualization in computer graphics and for performing some metrologic measurements. However, it does not convey high-level information on the geometry of the surface in the neighborhood of a point. Differential geometry is a popular approach for describing a surface. In differential geometry, the coefficients of the first and second fundamental forms of a surface patch *σ(u,v)* on a surface S in a differential neighborhood of a point P completely describe its intrinsic and extrinsic properties and, ultimately, its shape **[**DOCA1976]. As shown in Fig.3, given a parameterization *(u,v)*, a differential surface patch *σ(u,v)* at a point *P* has a surface normal N_s that is orthogonal to the tangent plane Π_t at *P*. A tangent vector *v* in Π_t can be expressed as a linear combination of σ_u and σ_v. Defining the linear maps $du(v) = \lambda$ and $dv(v) = \mu$, we obtain Eq. (5).

$$v = \lambda\sigma_u + \mu\sigma_v \qquad (5)$$

Applying the inner product $\langle \ \rangle$ to vector v and using Eq. (5) yields Eq. (6).

$$\langle v,v\rangle = \lambda^2\langle\sigma_u,\sigma_u\rangle + 2\lambda\mu\langle\sigma_u,\sigma_v\rangle + \mu^2\langle\sigma_v,\sigma_v\rangle \qquad (6)$$

Writing $E = \|\sigma_u\|^2$, $F = \sigma_u \square \sigma_v$, *and* $G = \|\sigma_v\|^2$ and using the maps $du(v)$ and $dv(v)$ above, the expression for $\langle v,v\rangle$ writes as Eq. (7).

$$\langle v,v\rangle = Edu^2 + 2Fdu\,dv + Gdv^2 \qquad (7)$$

Eq. (7) is referred to as the *First Fundamental Form I* of the surface. In an infinitesimal neighborhood of P, *I* describes the measurement of a length on the surface. Although *E*, *F*, *G*, σ_u and σ_v depend on the parameterization of the surface, the first fundamental form *I* depends only on *S* and *P*. It is an intrinsic property of the surface since it is independent of how the surface is embedded in 3D space. This can be better understood by visualizing the distance between two points on a flat sheet of paper. When the sheet is bent (without being folded), the distance between the points remains the same. It is thus an invariant property of *S* and is not affected by rotations and translations.

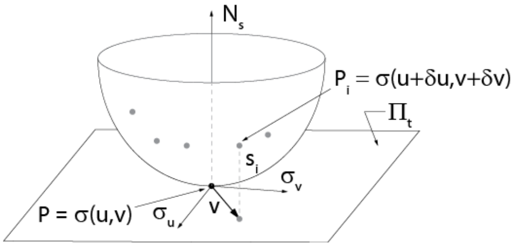


Figure 3. A small patch near a point P on surface σ(u,v) and the tangent plane Π_t

The *Second Fundamental Form II* of S describes the extrinsic properties of the surface around a point and is linked to the *curvature* of this surface, i.e. the way the surface pulls away from the tangent plane *Π_t* at *P*. Considering again Fig.3, it can be seen that the surface at point P_i pulls away from the tangent plane at P by a distance given in Eq. (8).

$$\left\langle \left(\sigma\left(u+\Delta u, v+\Delta v\right) - \sigma\left(u,v\right)\right), N_s \right\rangle \quad (8)$$

Approximating $\left(\sigma\left(u+\Delta u, v+\Delta v\right) - \sigma\left(u,v\right)\right)$ by its Taylor expansion and neglecting the high order terms, one obtains Eq. (9).

$$\sigma_u \Delta u + \sigma_v \Delta v + \frac{1}{2}\left(\sigma_{uu}\left(\Delta u\right)^2 + 2\sigma_{uv}\Delta u \Delta v + \sigma_{vv}\left(\Delta v\right)^2 \right) \quad (9)$$

Since σ_u and σ_v are tangent to the surface and are thus perpendicular to N_s, Eq. (9) becomes Eq. (10)

$$\frac{1}{2}\left(L\left(\Delta u\right)^2 + 2M \Delta u \Delta v + N\left(\Delta v\right)^2 \right) \quad (10)$$

with $L = \sigma_{uu},\ M = \sigma_{uv},\ N = \sigma_{vv}$. For small Δu and Δv, Eq. (10) can be written as Eq. (11) (if ½ is dropped)

$$\Pi = Ldu^2 + M\, du\, dv + Ndv^2 \quad (11)$$

Eq. (11) is called the *Second Fundamental Form II* of the surface at P.

The Shape Operator, also called the Weingarten Map, S in Eq. (12) can be defined at a point using the coefficients of the first and second fundamental forms [DOCA1976].

$$S = \left(EG - F^2 \right)^{-1} \begin{bmatrix} LG - MF & MG - NF \\ ME - LF & NE - MF \end{bmatrix} \quad (12)$$

The eigenvectors of S determine the directions in which the surface bends at each point and the eigenvalues κ_1 and κ_2 are the principal curvatures (i.e. the maximum and minimum normal curvatures at the point). It is possible to compute two very important invariant surface properties of a surface at a point: the Mean curvature *H* and the Gaussian curvature *K*. *H* and *K* are defined in Eq. (13) and (14) respectively.

$$H = \frac{\kappa_1 + \kappa_2}{2} \quad (13)$$

$$K = \kappa_1\ \kappa_2 \quad (14)$$

Although Eq. (13) and (14) are useful, it is more convenient to use the coefficients of *I* and *II* to compute H and K. The Gaussian curvature *K* is given by Eq. (15) while the Mean curvature *H* is given by Eq. (16) [PRES2010].

$$K = \frac{LN - M^2}{EG - F^2} \quad (15)$$

$$H = \frac{LG - 2MF + NE}{2(EG - F^2)} \quad (16)$$

Looking at Eq. (15) and (16), *K* and *H* can be obtained from differentials. Using the signs of K and H, it is also possible to characterize the type of surface to which a 3D point on a surface belongs to [BESL1988]. As shown in Table 1, seven types of surface can be described by the combination of the signs of K and H.

The usual approach that was proposed for finding the type of surface at a given point consisted in fitting a quadratic surface model in the *N x N* neighborhood of each point in a smoothed range map and then in computing the partial derivatives needed to extract K and H [BESL1988]. A range map is a 3D image defined as in Eq. (17) for which the surface parameterization is such that there is a depth value z corresponding to a coordinate pair (x,y) in a plane. The connectivity between 3D points in a range map is thus known compared to a point cloud for which the connectivity between points is unknown.

$$z(x, y) = f(x, y) \quad (17)$$

Although this fitting approach can achieve good results, computing the derivatives on the raw depth map (i.e. without fitting) is unpractical because of sensor noise. For large depth maps or, in a more general case for large point clouds, the fitting step is very time consuming. In addition, a different fit is implemented at each point even when points lie in the same neighborhood and may belong to the same surface type. However, as pointed out in [BESL1988], correcting this may require a priori assumptions on the surface type. Making such assumptions is very restrictive and does not allow generalization of the approach. When each point has been labelled with a given surface type, it is possible to group connected points sharing the same label and to fit a high-order polynomial (or spline model) to the region in order to obtain a high-level model.

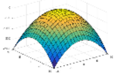
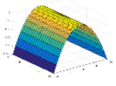
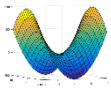
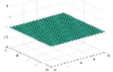
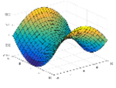
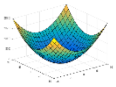
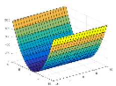
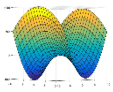
		K		
		+	0	-
H	-	Peak 	Ridge 	Saddle Ridge 
	0	none	Flat 	Minimal surface 
	+	Pit 	Valley 	Saddle Valley 

Table 1. Types of surface as a function of the signs of the Gaussian and Mean Curvatures

Covariance-Based Differential Geometry

A different approach to exploit differential geometry for finding the surface type at each point of a range map consists in using covariance-based differential geometry [BECK1994]. Returning to Fig.3, one can compute a local covariance matrix C$_I$ at point P of a range map as Eq. (18).

$$C_I = \frac{1}{N}\sum_{i=1}^{N}\left(\underline{P}-\underline{P}_m\right)\left(\underline{P}-\underline{P}_m\right)^t \quad (18)$$

P$_m$ is defined as in Eq. (19) where $\underline{P}_i$ is a point in the neighborhood of P on the range map. It is assumed that N points are selected in the neighborhood of P.

$$\underline{P}_m = \frac{1}{N}\sum_{i=1}^{N}\underline{P}_i \quad (19)$$

As described in [BECK1994], the eigenvectors of C$_I$ are three orthogonal vectors, two of which, t$_1$ and t$_2$, lie on the tangent plane to the surface at P (plane Π$_t$ in Fig.3) and the third one, corresponding to the smallest eigenvalue of C$_I$, is the normal Ns to the tangent plane (and the surface) as suggested in [LIAN1990]. In [BECK1994], the two-dimensional covariance matrix in Eq. (20) is defined. In Eq. (20), W$_i$ is a two-dimensional vector defined as in Eq. (21) with s$_i$ being defined as in Eq. (22).

$$C_{II} = \frac{1}{N}\sum_{i=1}^{N}\left(W_i - W_m\right)\left(W_i - W_m\right)^t \quad (20)$$

$$W_i = s_i\begin{bmatrix}\left(P_i - P\right)^t\ t_1\\ \left(P_i - P\right)^t\ t_2\end{bmatrix} \quad (21)$$

$$s_i = \left(P_i - P\right)^t\ N_s \quad (22)$$

As shown in Fig.4, vector W$_i$ is thus the difference between a point P$_i$ in the neighborhood of P projected on the vectors in the tangent plane weighted by the distance s$_i$ between P$_i$ and the tangent plane Π$_t$ at P.

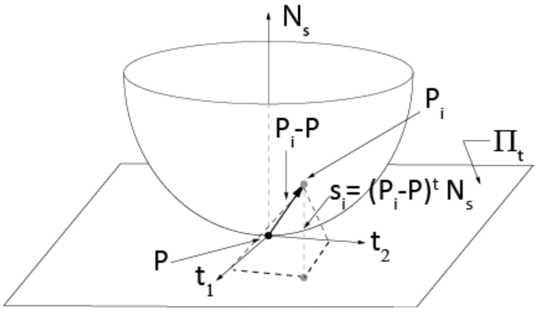


Figure 4. Geometry for the vectors in Equations (16), (17) and (18)

Beckman *et al.* define the quadratic form in Eq. (23) as a “covariance-based Weingarten map” for a vector v in the tangent plane.

$$II_C = \underline{v}^t\ C_{II}\ \underline{v} \quad (23)$$

Beckman *et al.* claim that the eigenvectors of C$_{II}$ are the principal directions on the surface, i.e. the directions of minimum and maximum normal curvature. They also define a covariance-based approach analogous to the Gauss map at a point P of the range map as in Eq. (24) with vector ν$_i$ defined as in Eq. (25).

$$C_P = \frac{1}{N}\sum_{i=1}^{N}\left(\nu_i - \nu_m\right)\left(\nu_i - \nu_m\right)^t \quad (24)$$

$$\nu_i = \begin{bmatrix} n_i^t\ t_1\\ n_i^t\ t_2\end{bmatrix} \quad (25)$$

The 2 x 2 matrix C$_P$ in Eq. (24) is the covariance matrix of the projections of the normal vector n$_i$ at points P$_i$ in the neighborhood of P, ν$_m$ being the average vector of the projections. The eigenvectors of C$_P$ are the principal directions. The eigenvalues of C$_P$ provide information on the way the surface normal in the neighborhood of P projects onto the tangent plane. For instance, if the surface in the neighborhood of P is a plane, the normal vectors all map into a single point,

point P itself. When one eigenvalue is large and the other is small, the projection of the surface normal vectors map on a straight line and the underlying surface is a developable parabolic surface. Finally, when both eigenvalues are large, the surface is locally curved near P. Beckman et al. have applied the above covariance-based approach to segment points in a range map. In comparison with pure differential geometry approaches such as the one presented in [BESL1988], Beckman's covariance-based approach can only identify three different types of surface: planar, parabolic and curved. A planar surface is identical to the "flat" surface type (with K = H = 0) in Table 1. A parabolic surface covers the cases of ridge (K=0, H<0) and valley (K=0, H>0) in Table 1 while a curved surface covers the other in Table 1. Consequently, the segmentation obtained by covariance-based differential geometry is less rich than the one obtained with classical differential geometry for the reason that, as demonstrated in [DIGN2014], even though the eigenvectors of C_P correspond to the principal directions, the eigenvalues of C_P are not equal to the principal curvatures κ1 and κ2 but are rather functions of their squared value as expressed in Eq. (26) and Eq. (27) (where r is the radius of the ball centered at P).

$$\lambda_{1-C_P} = \frac{\kappa_1^2 \, r^4 \pi}{4} + o\left(r^4\right) \qquad (26)$$

$$\lambda_{2-C_P} = \frac{\kappa_2^2 \, r^4 \pi}{4} + o\left(r^4\right) \qquad (27)$$

Because of this, it is not possible to find the sign of H and, consequently, to differentiate between ridge or valley or peak / pit in Table 1.

The following sections explain how these limitations can be circumvented and how the 3D data can be segmented into the seven surface types in table 1(here, minimal surface is considered as a saddle).

3. PROPOSED APPROACH

As mentioned previously, the sign ambiguity of the eigenvalues obtained by C_{II} prevents us from distinguishing some surface types. We propose a new technique that combines covariance-based differential geometry and the vector field implicit surface representation to segment the 3D data. Instead of working with each point and the neighborhood around that point, we rather work with the voxels in the volumetric grid containing the vector field and its 26 possible neighbors in the grid.

As mentioned above, handheld scanners for metrologic applications use retroreflective markers or natural features to estimate the pose of the sensor with respect to a reference frame chosen as the "world" reference frame. Secondly, using the vector field implicit surface representation, view registration, view integration and the estimation of the normal to the surface in each voxel of the field can be performed in real time as the 3D data is collected by the handheld sensor. The vector field is also built in the world reference frame. Since the pose of the sensor in the world reference frame is estimated in real-time, it is also possible to know on which side of the surface the sensor is when 3D data is collected and integrated in a voxel of the vector field. Knowing on which side of the surface the sensor is located when the data is collected and the surface normal in the voxel allows the orientation of this normal to be defined with respect to the direction of the optical axis of the sensor. This also allows the differentiation between peak/pit or ridge/valley and eliminates the limitations of covariance-based differential geometry (expressed by Eq. (26) and Eq. (27)) for surface segmentation. Based on the above, the strategy that is proposed for surface segmentation is to apply covariance-based geometry on the vector field implicit surface representation instead of on individual points, thus reducing the computational load considerably since hundreds of points if not thousands fall in a single voxel.

In the context of the differential geometry, when the orientation of the normal vector is given, then it is possible to observe that if

- The distance between the tangent plane and all of the points of a certain region around the particular point are positive, then the point is a peak surface.
- The distance between the tangent plane and all of the points of a certain region around the particular point are negative, then the point is a pit surface.
- The distance between the tangent plane and all of the points of a certain region around the particular point is both positive and negative, then the point is a saddle surface.

- The distance between the tangent plane and all of the points of a certain region around the particular point is both positive and zero, then the point is a ridge surface.
- The distance between the tangent plane and all of the points of a certain region around the particular point is both negative and zero, then the point is a valley surface
- The distance between the tangent plane and all of the points of a certain region around the particular point is all zero, then the point is a plane.

We extend this concept into the vector field framework. Therefore, instead of using points we rather use a voxel and the neighbours around the voxel. With the additional knowledge of the direction of the surface normal (available in each voxel of the field), it is possible to exploit Eq. (22) on the neighborhood of a voxel to identify to which type of surface the points in this voxel belong to.

Implementation

Since the covariance matrix of the points falling inside a voxel is computed in real time, computation of the normal vector is also achieved in real time and is the eigenvector of the covariance matrix corresponding to the smallest eigenvalue. By implementing the vector field framework, the normal vector, the closest point to the surface (Eq. (3)), the sensor position and the voxel center are all stored in a voxel of the 3D grid. Therefore, all of the values needed to compute the orthogonal distance from the tangent plane of a particular voxel to the neighboring voxels are provided. For each voxel in the 3D volumetric grid, there are 26 possible neighbors. We have to consider the neighboring voxels which contain much more than 3 points inside in order to have a reliable covariance matrix. This is not a problem since modern scanners can capture 250,000 points per second. We define the orthogonal distance form as

$$Distance_j = (c_v(j) - c_v(0))^T.n \qquad (28)$$

where j = 1 to 26 is the number of the neighbouring voxels around the voxel v_0 . $c_v(j)$ is the point on the plane approximating the surface in the jth neighboring voxel (point c_v in Figure 2), $c_v(0)$ is the point on the plane approximating the surface in the voxel of interest, n is the normal vector obtained from the covariance matrix in the voxel of interest of the vector field framework. $Distance_j$, is a $j \times 1$ vector stored in each voxel of the vector field framework. An illustration in 2D to simplify visualization is given in Fig. 5.

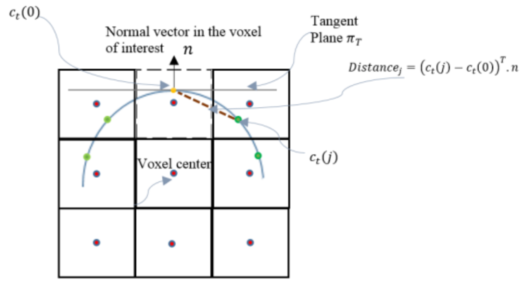

$c_t(0)$
Normal vector in the voxel of interest n
Tangent Plane π_T
$Distance_j = (c_t(j) - c_t(0))^T.n$
$c_t(j)$
Voxel center

Figure 5. An illustration of $Distance_j$ stored in a voxel of interest.

The information of this matrix allows us to recognize the surface type in a given voxel as follow:

- If all of the values of the $Distance_j$ are positive, then the surface in the particular voxel is a peak surface.
- If all the values of the $Distance_j$ are negative, then the surface in the particular voxel is a pit surface.
- If all of the values of the $Distance_j$ are both positive and negative, then the surface in the particular voxel is a saddle surface.
- If all of the values of the $Distance_j$ are both positive and negative, and the eigenvalues obtained by Eq.(26), Eq. (27) are equal then the surface in the particular voxel is a minimal surface. Which, in this paper are considered as belonging to the same category as the saddle surfaces.
- If all of the values of the $Distance_j$ is both positive and zero, then the surface in the particular voxel is a ridge surface.
- If all of the values of the $Distance_j$ is both negative and zero, then the surface in the particular voxel is a valley surface.
- If all of the values of the $Distance_j$ are zero, then the surface in the particular voxel is planar.

No filtering or fitting operation are performed on the data.

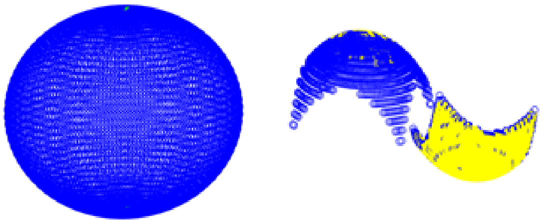
4. EXPERIMENTAL RESULTS

In this section, the experimental results are presented to demonstrate the performance of our approach. The method is applied to 3D synthetic data: Plane, Sphere (peak, pit), Cylinder (Valley, Ridge) and Saddle surface, as well as 3D data which was obtained from a real scanner (Stanford repository) and a HandyScan 3D Laser Scanner by Creaform. The following experiments validate our method. The color map for different surface types is shown in Table 2.

Surface Type	Colour list
Pit surface	Yellow
Ridge surface	Green
Valley surface	Black
Saddle and Minimal surface	Cyan
Peak surface	Blue
Plane surface	Red

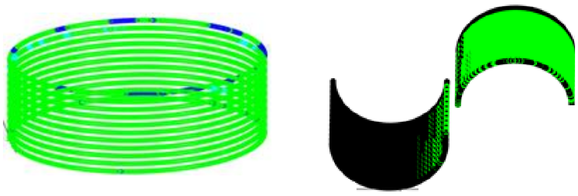
Table2. Color map corresponding to different surface types

Fig.6 to Fig.14 show the result of the voxels segmented into different surface types. The color in the voxels is coherent with the surface types in each voxel. The results on synthetic spherical surfaces (peak in blue and pit in yellow) and the results on synthetic cylindrical surfaces, which are valley (black) and ridge (green) surface are shown in Fig.6 and Fig.7 respectively.



(a) (b)

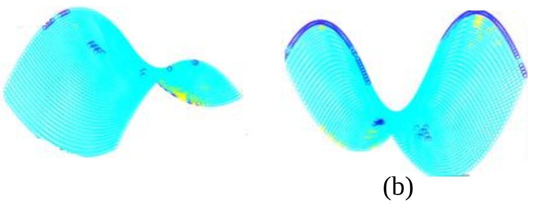
Figure 6. The result of the proposed covariance differential geometry segmentation on a synthetic sphere. (a) Shows peak in blue. (b) Shows peak in blue and pit in yellow color



(a) (b)

Figure 7. The result of the proposed covariance differential geometry segmentation on a synthetic cylinder. (a) Shows the ridge surface in green (b) Shows the ridge surface in green and the valley surface in black color.

Fig.8 shows the result of our approach on synthetic saddle and minimal surface, in this paper we classify minimal surfaces in the same category as the saddle surfaces because they are a special case of saddle for which the $|k_1| = |k_2|$. So both are shown in the same color.



(b)

Figure 8. The result of the proposed covariance differential geometry segmentation on a synthetic saddle surface. (a) Shows the minimal surface (in this paper considered as saddle) in cyan (b) Shows the saddle surface in cyan color

Fig.9 is an example of segmentation of a planar surface.

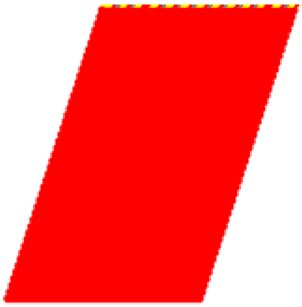
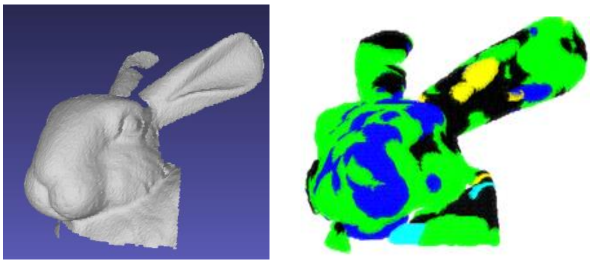


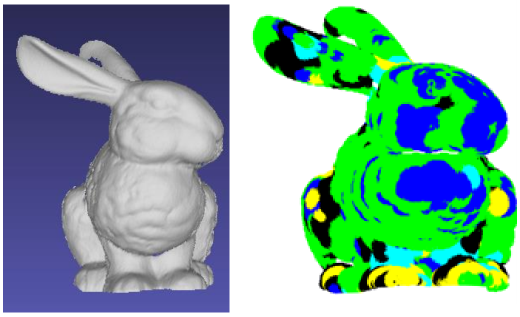
Figure 9. Result of the covariance differential geometry segmentation on a synthetic planar surface

Fig.10 shows the segmented regions in a bunny's head. One can observe on the model data the regions around the muzzle and cheek are pit and the regions around the ears are mostly cylindrical (valley and ridge). So as expected, Figure 9 (b) shows the region around muzzle and cheek in blue and voxels around the ear in green and black which prove the efficiency of our approach.



(a) (b)

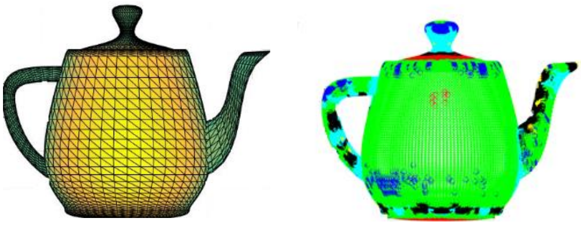
Figure 10. The result of the proposed covariance differential geometry segmentation on a 3D point provided by the Stanford repository. (a) Object mesh data (b) The segmented regions on 3D points of bunny's head



(a) (b)

Figure 11. The result of the covariance differential geometry segmentation on a 3D point provided by Stanford repository. (a) Object mesh data. (b) The result of the segmentation on the 3D points

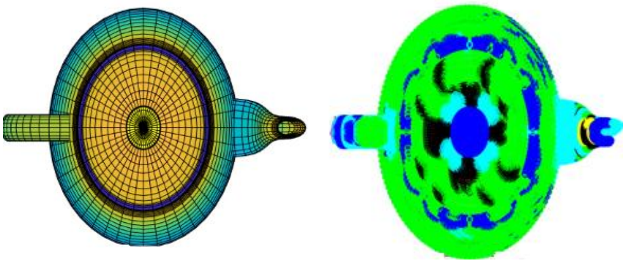
By referring to Fig.11 (a), it can be observed that the upper side of the bunny's belly is peak surface and under the belly is more ridge and the regions around the claws of the bunny are mostly pit and valley. Clearly the regions betweens the belly and claws are saddle, and Fig.11 (b) provide a qualitative validation of our approach on the 3D points of the bunny.



(a) (b)

Figure 12. The result of the covariance differential geometry segmentation on a teapot. (a) Object mesh data. (b) The result of the segmentation on 3D points

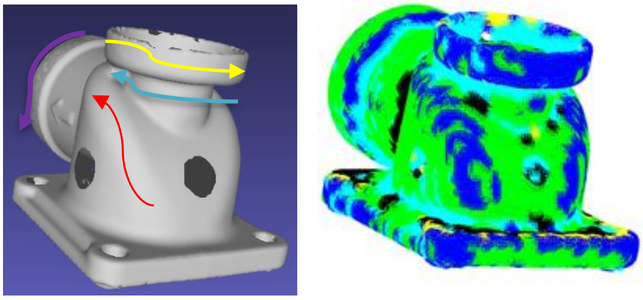
It is apparent that for the teapot the main body is basically cylindrical. And it is also clear that the knob in the teapot is peak and the spout is a saddle surface. Fig. 12 and Fig.13 show the main body in green color which is a ridge surface and the knob in blue colour which demonstrate that the region is peak as expected. The spout in cyan color is also coherent with saddle surfaces.



(a) (b)

Figure 13. The result of the covariance differential geometry segmentation on a teapot. (a) Object mesh data. (b) The result of the segmentation on 3D points

The object shown in Fig.14 was scanned by a HandyScan scanner by Creaform. As labeled in Fig.14 (a) the red arrow shows the region that are curved and is a mixture of peak and ridge surfaces. The region shown by purple arrow is ridge. The blue arrow corresponds to a saddle surface. The regions on the object shown by the yellow arrow are peak. We obtain corresponding segmentation in Fig.14 (b) as expected.



(a) (b)

Figure 14. The result of the covariance differential geometry segmentation on a 3D point collected by HandyScan (Creaform) scanner. (a) Object mesh data. (b) The result of the segmentation on the 3D points

Performance Evaluation

To evaluate the performance of the proposed method, a comparison between quadratic fitting at each point of the point cloud and the proposed method using the vector field has been done. In this section, the details of estimating the performance timing in MATLAB (R2016a) installed on the system with CPU (Intel(R)Core(TM) i7-5820K CPU @ 3.30 GHz 3.30 GHz) and memory (RAM 48.0 GB) for the "teapot" object is presented. The total number of 3D points for the "teapot" object is 41472 points. To evaluate the performance time, we sample 20 points in the neighborhood of a point of the teapot. After sampling, a quadratic equation was fitted on the sampled points. The execution time was "37.3792" seconds for the 20-point neighborhood. For a total number of points of 40000 points, the estimated time to perform the quadratic fitting would be around 74000 seconds. On the other hand, the time required to run the proposed method on the vector field representation using covariance-based differential geometry is 1027.6 seconds, which is much shorter than the performance time for quadratic fitting on the points. This performance is achieved for a vector field grid composed of 35991 voxels with 23661 non-empty voxels. Only the non-empty voxels are processed.

5. CONCLUSION and FUTURE WORK

In this paper, we revisit the concepts of differential geometry used for the segmentation of range maps into different surface types in the more recent context of implicit surface representation and demonstrate that differential geometry can be used efficiently for surface segmentation without the need of surface fitting or the estimation of derivatives.

The paper also extends the concepts used for range maps to unorganized point clouds. There is a significant advantage of combining covariance-based differential geometry approaches and the vector field framework since the segmentation does not require surface fitting. The key point in our approach is that instead of working with points, we are working with voxels and their neighbors, which reduces the computational complexity. The future work of our approach is choosing a higher-level reconstruction method for the surface in the segmented regions, then investigating the continuity between the segmented voxels and their neighbors which do not require high order surface representation.

6. ACKNOWLEDGMENTS

The authors acknowledge the support of the NSERC/Creaform Industrial Research Chair on 3-D Scanning for conducting the work presented in this paper.

7. REFERENCES

[BECK1994] J. Beckman, T. Caelli, “Computation of Surface Geometry and Segmentation Using Covariance Techniques,” IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 16, no. 11, Nov. 1994, pp. 1114-1116

[BESL1988] P. Besl, R.C Jain, “Segmentation Through Variable-Order Surface Fitting,” IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 10, no. 2, March 1988, pp. 167-192

[CHEN2012] S.-W. Cheng, T.K. Dey, J.R. Shewchuk, “Delauney Mesh Generation,” Chapman & Hall / CRC computer and information science series, 2012, 386 p.

[CURL2996] B. Curless, M. Levoy, “A volumetric method for building complex models from range images,” SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, August 1996 Pages 303–312

[DANE2018] M. Daneshmand, A. Helmi, E. Avots, F. Noroozi, F. Alisinanoglu, H. Sait Arslan, J. Gorbova, R. E. Haamer, C. Ozcinar, G. Anbarjafari, “3D Scanning: A Comprehensive Survey,” Jan. 2018, arXiv:1801.08863v1

[DIGN2014] J. Digne, J.M. Morel, “Numerical Analysis of Differential Operators on Raw Point Clouds,” Numerische Mathematik, (2014), 127:255-289, DOI 10.1007/s00211-013-0584-y

[KHAK 2019] H. Khaksari-Haddad, D. Laurendeau,” Alignment of Point Clouds for Comparison of Infrastructures in Engineering on Quality Control”, Journal of WSCG,Vol 27, no1, October 2019.

[DOCA1976] M.P. do Carmo, “Differential Geometry of Curves and Surfaces,” Prentice-Hall, © 1976, 503 p.

[LIAN1990] P. Liang and J. S. Todhunter, “Representation and recognition of surface shapes in range images,” Computer Vision, Graphics and Image Processing, vol. 52, no. 10, pp. 78-109, 1990

[LORE1987] W.E. Lorensen, H.E. Cline, “Marching cubes: A high resolution 3D surface construction algorithm,” SIGGRAPH ’87 Conference Proceedings, 21(4):163–169, 1987

[PRES2010] A. Presley, “Elementary Differential Geometry,” Springer, 2010, 468 p.

[RUSI2001] S. Rusinkiewicz, M. Levoy, “Efficient variants of the ICP algorithm,” Proceedings Third International Conference on 3-D Digital Imaging and Modeling (3-DIM), 28 May-1 June 2001, DOI: 10.1109/IM.2001.924423, pp. 145-152

[TUBI2002] D. Tubić, P. Hébert, D. Laurendeau, « A volumetric approach for interactive 3d modeling,” In Proceedings of First International Symposium on 3D data Processing Visualization and Transmission (3DPVT), volume 1, June 2002, pp. 150–158.

Upper Airway Segmentation using Fast Marching

César Bustacara-Medina
Pontificia Universidad Javeriana
Bogotá D.C., Colombia
cbustaca@javeriana.edu.co

Leonardo Flórez-Valencia
Pontificia Universidad Javeriana
Bogotá D.C., Colombia
florez-l@javeriana.edu.co

José Hernando Hurtado
Pontificia Universidad Javeriana
Bogotá D.C., Colombia
hhurtado@javeriana.edu.co

ABSTRACT

Direct measurements of airway tree and wall areas are potentially useful as a diagnostic tool and as an aid to understanding pathophysiology underlying of the airway diseases. Direct measurements can be made from images obtained using computer tomography (CT) by applying computer-based algorithms to segment airway, however, current validation techniques cannot establish adequately the accuracy of these algorithms. Additional, the majority of the studies only include the airway from trachea to bronchi's tree avoiding the upper respiratory system, because the main problems appears in the lower respiratory system, for example, asthma and chronic obstructive pulmonary (airflow obstruction or limitation, including chronic bronchitis, emphysema and bronchiectasis). Airway tree segmentation can be performed manually by an image analyst, but the complexity of the tree makes manual segmentation tedious and extremely time-consuming (require several hours of analysis), only including trachea and lower airway system. Airway segmentation in CT images is a challenging problem for two reasons, it is a complex anatomy and exists limitations in image quality inherent to CT image acquisition. This paper describes a semi-automatic technique to segment the airway tree (upper airway system and trachea), using CT images of head-neck and applying a fast marching algorithm. Additionally, a heuristic is proposed to determine the algorithm parameters without have to review manually all structures to segmentation.

Keywords

Image segmentation, Airway tree, Fast marching, Computed tomography.

1. INTRODUCTION

Image analysis techniques have been broadly used in computer-aided medical analysis and diagnosis in recent years [3][4][5][6][7]. Computer-aided image analysis is an increasingly popular tool in medical research and practice, especially with the increase of medical images in modality, amount, size, and dimension. Image segmentation, a process that aims at identifying and separating regions of interests from an image, is crucial in many medical applications such as localizing pathological regions, providing objective quantitative assessment and monitoring of the onset and progression of the diseases, as well as analysis of anatomical structures [8][9][10][11][12]. A number of techniques have been developed for segmenting and analyzing the 3-D airway tree. Kitasaki *et al.* [13] used a voxel classification based on local intensity structure. Hirano *et al.*[14] used the cavity enhancement filter and the region-growing method. Park *et al.* developed an integrated software package utilizing a new measurement algorithm called mirror-image Gaussian fit (MIGF), that enables the user to perform automated bronchial segmentation, and measurement. In addition, MIGF permits to delineate the outer wall of bronchia. Bauer [16] used Gradient Vector Flow (GVF) method to produces accurate segmentation of the airway lumen. Aykac et *al.* [1] used grayscale morphological reconstruction to identify candidate airways. Park et *al.* [17] used 3-D confidence connected region growing (CCRG) method to extract lower and upper airways of the bronchi.

Many researchers agree that the complex branching structure of the human airway tree can be examined using computed tomography (CT) imaging. Quantitative analyses can be performed on the three-dimensional (3D) airway tree to evaluate tree structure and function [2][18][14][15][17]. It is important to note that most of segmentations are performed of the trachea and the lower airway tree, but in our case, the segmentation of the upper airway and trachea is necessary, since the results will be used to characterize the possible causes of sleep apnea in a series of 387

https://doi.org/10.24132/CSRN.2020.3001.21

patients.

In this paper we will use the level sets technique proposed for Osher and Sethian in 1988 [19], specifically, the fast marching method introduced by Sethian in 1995 [20][21]. As shown by Sethian, fast marching is a set of finite difference numerical techniques that were constructed to solve the Eikonal equation, which is a boundary value partial differential equation. These techniques rely on a marriage between the numerical technology for computing the solution to hyperbolic conservation laws and the causality relationships inherent in finite difference upwind schemes. Fast marching methods are Dijkstra type methods, in that they are closely connected to Dijkstra's well-known network path algorithms [22].

2. MATERIALS

Our proposed method has been applied to 70 patients. By each patient, two CT scan were acquired, one in awake and other in induced sleep. The dataset include the head and thorax, because we are interested in the upper airway and the trachea, specifically, from the lower trachea to the nasal conchae as illustrated in Fig. 1. The paranasal sinus region contains several membrane-like structures, affected by partial volume effects, which renders the segmentation as a non-trivial process. Additionally, poor contrasts are usually caused by existing metal parts (tooth implants or other implants) that appear in extremely high intensities relative to the actual anatomical structures. The image size of all patients dataset is 512×512 pixels in each slice, 460 - 840 slices per image, 0.368-0.586 [mm] in pixel spacing, and 0.299-0.301[mm] in slice spacing, respectively.

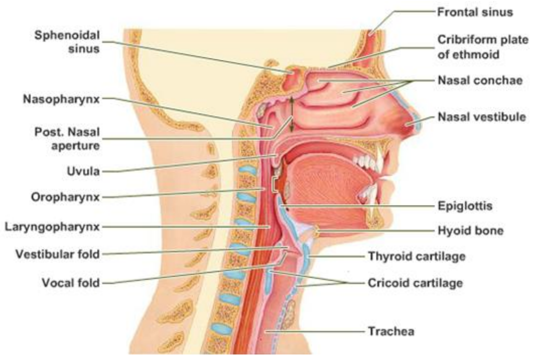


Fig. 1 Upper Airway Anatomy, adapted from [23]

The fast marching segmentation method is implemented in the Insight Segmentation and Registration Toolkit (ITK) and can be downloaded from www.itk.org [24]. ITK was funded by the National Library of Medicine (NLM), and it is open-source software that was developed jointly by six principal organizations to support the Visible Human project of NLM. ITK includes several basic segmentation and registration techniques that have been implemented for a variety of medical image analysis applications. Additionally, the visualization was made using the Visualization Toolkit (VTK).

3. METHOD

Fast marching methods are finite difference techniques, more recently extended to unstructured meshes, for solving the Eikonal equation of the form[21]:

$$|\nabla T| F(x, y, z) = 1, T = 0 \text{ on } \Gamma \qquad (1)$$

This can be thought of as a front propagation problem for a front initially located at $\Gamma$and propagating with speed $F(x, y, z,) > 0$. The evolution of the function is controlled by a partial differential equation in which a speed term F is involved. In our case, we use the fast marching filter implemented in ITK as illustrated in Fig. 2.

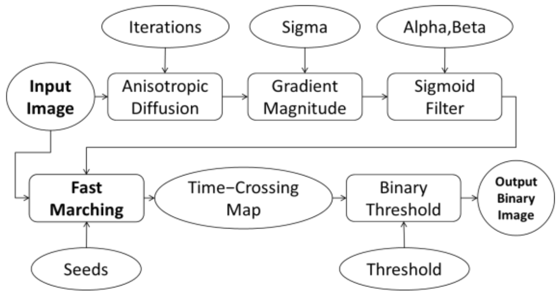


Fig. 2 Fast marching algorithm in ITK

A typical speed image is produced by mapping of gradient magnitude of the original image. The mapping is selected in such a way that regions with high contrast will have low speeds while homogeneous regions will have high speed. In this case the mapping of gradient magnitude is made with a sigmoid function. The fast marching filter will propagate a front starting from the seed points defined for the user and traveling with the speed computed from the speed image. This should result in the contour slowing down close to object edges. The result of the filter is a time-crossing map which indicates for each pixel, how long it took to the contour to reach this pixel.

3.1 Gradient magnitude

Obtaining a good gradient magnitude is important to define the limits of the structures that you want to segment and thus be able to apply a sigmoid filter to obtain the speed function that the fast-marching algorithm will use. According to the above, it is necessary to determine the most appropriate algorithm and parameters to obtain the maximum gradient magnitude. These algorithms are associated with edge detection achieving as indicated by Canny and Deriche [25][26], applying an efficient criterion for edge detection, which allows a good detection, localization and good response to a single edge.

The good detection implies that there must be a very low probability of failure in detecting a real edge at some point and a low probability of scoring points that are not edges. This criterion as indicated by Canny [25], corresponding to maximize the signal-noise ratio (SNR), which means, find the maximum response of the detection criterion, which correspond to find the best operator to detect edges only.

To detect sudden intensity changes in the image two approaches have been used. The first, location of the extremes (maxima and minima) of the first derivative of the intensity function (image), and the second, location of the zero crossings or transitions from negative to positive values, or vice versa, from the second derivative of the intensity function (image). Besides these detectors based on the gradient or Laplacian, others have been proposed by optimizing certain criteria related with edge detection, the most popular algorithms are the Canny [25][27] and Deriche [26].

In this case, Deriche's algorithm which is based on the Canny will be used because proposes a robust formulation for the detection and location. In addition, using the Deriche-filter leads to a distortion of the amplitudes of the edges depending on their direction. The implementation presented by ITK for Deriche filter is called "RecursiveGaussianImageFilter" [24], the base class is defined to calculate the convolution of the IIR filter with a Gaussian kernel approach

$$\frac{1}{\sigma\sqrt{2\pi}}e^{-\left(\frac{x^2}{2\sigma^2}\right)} \qquad (2)$$

This class, according to ITK documentation, implements the Deriche's recursive filter proposed in his paper [26] and details or modifications presented by Deriche [28] and Farneback *et al.* [29]. Meanwhile, the magnitude of the gradient is calculated using the filter called "GradientMagnitudeRecursiveGaussianImageFilter", which convolved with the first derivative of a Gaussian function and invokes Deriche filter. In addition, the sigma (σ) parameter according with ITK is measured in spacing units of the image.

3.2 *Sigmoid function*

According to documentation provided by ITK [24], the sigmoid filter is commonly used as a transformation of intensity. Through this filter, for a specific range of intensity values is generated a new range of intensity obtaining a very smooth and continuous transition at the boundaries of the range. A sigmoid function is widely used as a mechanism to focus attention on a particular set of values and progressively reduce the values outside that range. It should be noted that this filter is a pixel operator, that is, for a given input generates a single output value.

The implementation of the sigmoid filter in ITK includes four parameters that can be adjusted to select their range of input and output intensity. Equation (3) represents the sigmoid transformation applied to each pixel of the image [24].

$$I' = (Max - Min) \cdot \frac{1}{\left(1+e^{\left(\frac{I-\beta}{\alpha}\right)}\right)} + Min \qquad (3)$$

The heuristic proposed by ITK to find the appropriate values to evaluate the sigmoid filter is: taking the image result to calculate the gradient magnitude, you must select the minimum value (K1) along the contour of the anatomical structure to be segmented. Then select the mean value (K2) of the gradient magnitude at the center of the structure. Under ideal conditions, K1 will always be greater than K2. This happens because K1 is located on the contour and K2 is located in the valley of the structure to be segmented. These two values indicate the dynamic range to be mapped to the interval [0, 1] in the resulting image, which will be used as the speed function for Fast Marching segmentation method. After other considerations, this mapping will produce a speed image such that the level set, it will march rapidly on the homogeneous region and it will stop on the contour. The suggested value for β is (K1+K2)/2 while the suggested value for α is (K2-K1)/6, which must be a negative number. In our case, determine the appropriate value for K1 is quite difficult and tedious because the target structure is elongated and presents complex regions (upper airway), requiring much time to explore the contours per-slice.

One important phase in the segmentation process is based on the adaptation (tuning) of gradient magnitude and sigmoid function parameters. The approach is based on define the parameters using visual information obtained from ideal behavior of the functions. This will reduce time-consuming due to the user-interaction required per-slice.

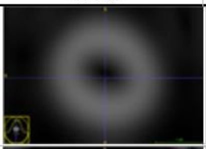
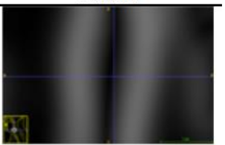
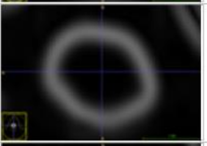
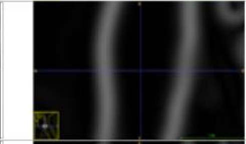
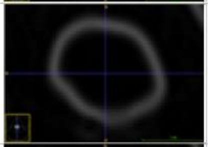
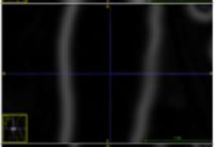
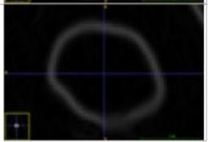
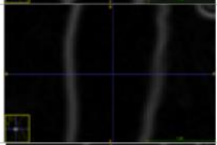
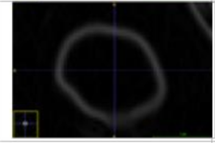
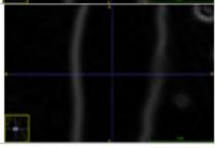
4. RESULTS

Pre-processing is first performed on the image, which includes denoising the dataset using an edge-preserving smoothing filter (e.g. Anisotropic Diffusion). The use of such smoothing is preferable since traditional blurring approaches tend to resolve thin, sharp structures that are important in our case. Then, "GradientMagnitudeRecursiveGaussianImageFilter" filter was applied to calculate the magnitude of the gradient.

As the objective is to determine the best value for sigma (σ), various tests were performed. Table I presents the results for a CT image (patient No. 1), varying the value of sigma between 3.0 and 0.01 (some intermediate results were omitted). As the value of sigma (σ) is reduced, the edges are appreciated more centered (focused) and localized, but this does not imply that they are of better quality (magnitude) to apply sigmoid filter and therefore obtain good segmentation of the desired region.

Since the maximum value of the gradient magnitude is locally obtained, the maximum and minimum values for the entire dataset (70 CT) were taken. Fig. 3 shows the behavior obtained by varying the sigma value between 3.0 and 0.01. The sigma value with which the maximum values for the gradient magnitude are obtained is 0.09, except in some cases whose maximum value is in sigma 0.1 and 0.08. These results are independent of the intensity range and spacing present in the images.

TABLE I
GRADIENT MAGNITUDE

σ	Axial	Coronal
3.0		
1.0		
0.5		
0.1		
0.09		

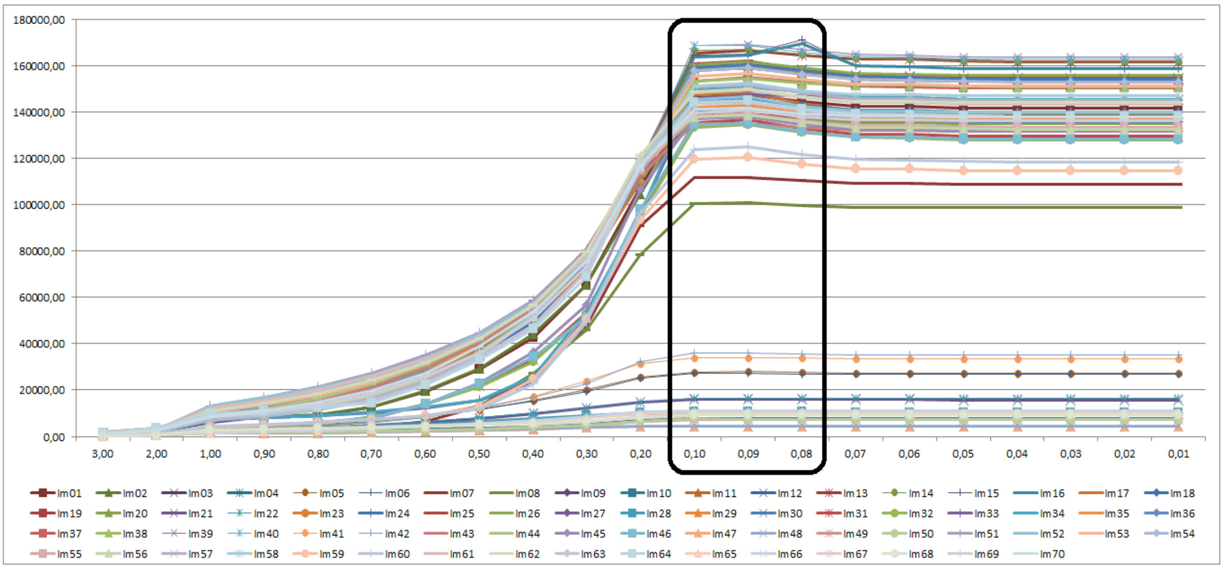


Fig. 3 Gradient magnitudes (70 CT)

Now, verification of obtained values from the gradient magnitude images is required, for this an axial slice of the 3D image located in the area of interest (airway) is taken. Fig. 4 illustrates the analyzed region between points 213-279. Values along the x-axis are extracted and then plotted as shown in Fig. 5. The value of gradient magnitude at the edge of the analyzed region is increased as the value of sigma is decreased. The maximum value at border points (edges at 228 and 262) is reached when sigma value is 0.09 and then remains stable.

From the above, we can conclude that the best sigma value is located between 0.1 and 0.09, so the tests of the sigmoid filter will be made with a sigma value of 0.09.

Sigmoid filter require values K1 and K2 that must be obtained from observing the gradient magnitude image. This is easier for regions to segment with certain characteristics, for example, small size and

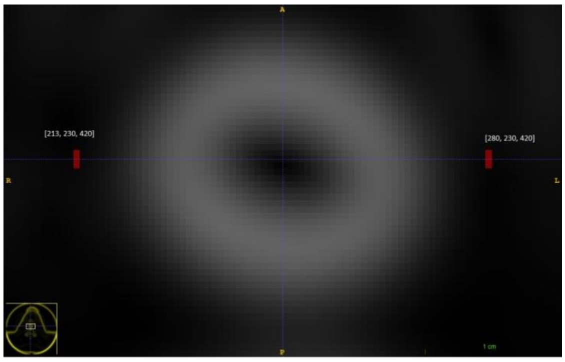


Fig. 4 Axial slice of the airways

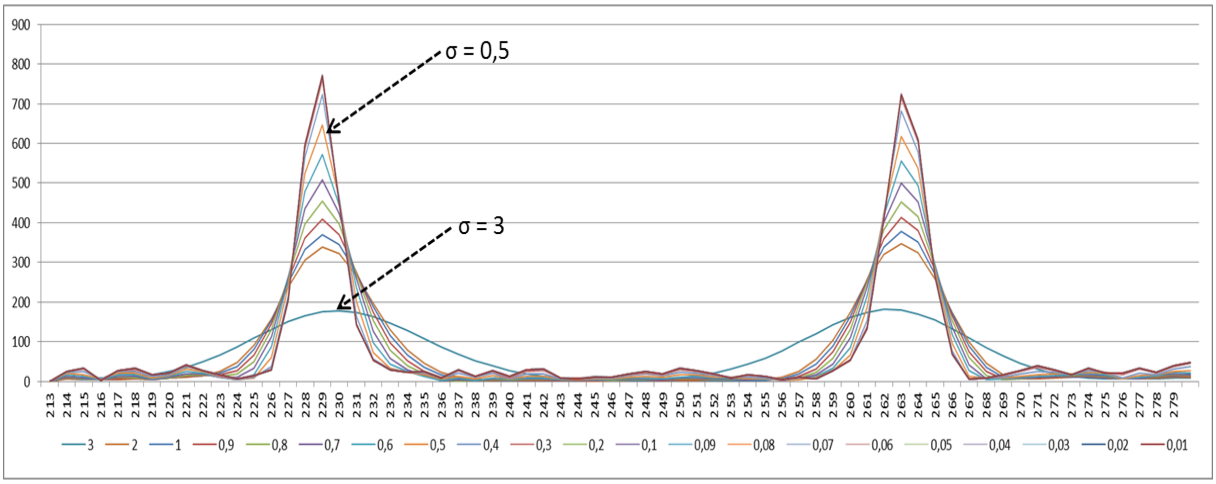


Fig. 4: Gradient magnitude in edges

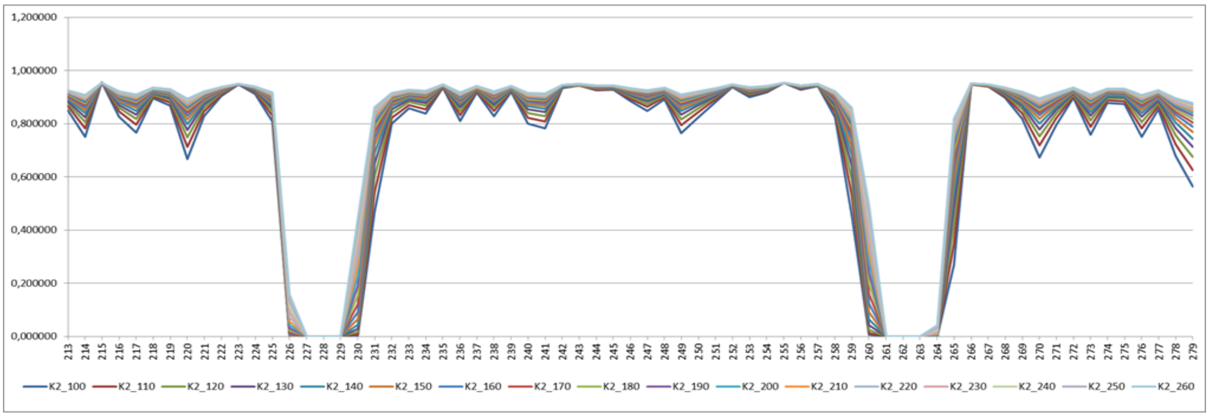


Fig. 6: Sigmoid behaviors

easy navigation to extract these values accurately. But unfortunately, the above is not manageable for long structures that occupy a lot of CT slices, such as airways or blood system. The sigmoid filter was applied to the gradient magnitude images with values to K1 between 100 and 500 and K2 between 4 and 10. Just as for gradient analysis, a slice in the region of interest was taken and the sigmoid behavior was plotted as shown in Fig. 6. To small values of K2 abrupt changes occur in the sigmoid function inside the structure. But according with the definition, it is expected that the speed function allows the propagation of level-set, thus, the speed function must go to zero smoothly on the structure boundaries, and this behavior was verified as shown in Fig. 7.

Several curves cross with x-axis, which can generate an over-segmentation (overflows the propagation front). The behavior of the sigmoid should be smooth and should not cross x-axis in the boundary of structure. Based on the above, the user can select the set of curves that best satisfy the described conditions, achieving thus determine the most appropriate values for K1 and K2. In our case, the values for K1 and K2 correspond to the interval 130-160 and 6-8 respectively.

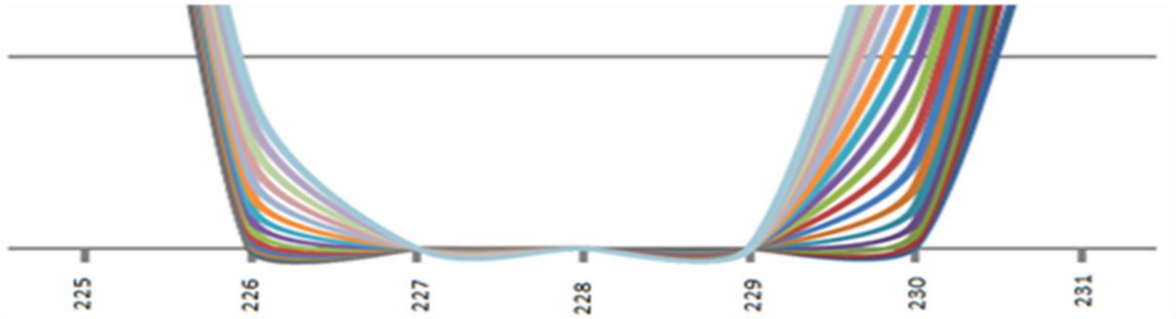


Fig. 7 Sigmoid behaviors on structure boundaries

Applying the fast marching algorithm with the parameters defined using the proposed heuristic, it is observed that as the parameter K1 is close to value 150, the results generated are closer to the real boundary of the desired structure. This imply that the proposed values are suitable for segmentation, as illustrated in Fig. 8. Further, as indicated, when the value of K1 (minimum value on the structure boundaries) is high, it is obtained an over-segmentation and vice versa a sub-segmentation.

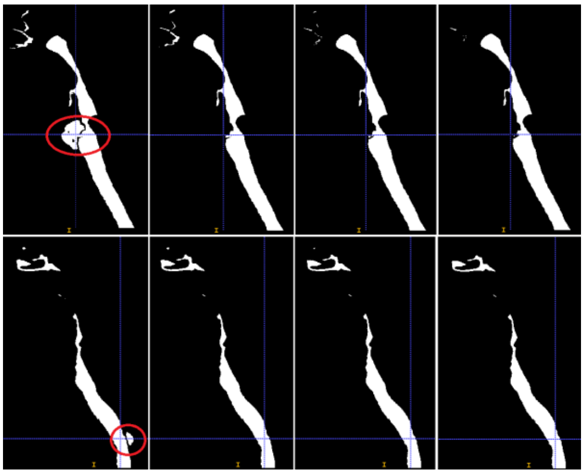


Fig. 6 Fast marching results for K1={170, 150, 140, 130} and K2=6.

With this new heuristic to determine the parameters to evaluate the gradient magnitude (K1, K2) and speed function defined for these values, the time-consuming to explore the structures to define K1 and K2 is less. This permit to eliminate the overall review of the structure to be segmented.

5. CONCLUSIONS

We have proposed a heuristic that can be split into two main stages: Gradient magnitude and speed function. The key point of the gradient magnitude is to use optimal recursive and separable filters as Deriche proposed to obtain approximate gradient or Laplacian. Select the maximum value reduces the possibility of fronts propagation overflow outside the structures to segment, which is associated with the second key point. The speed function can be defined using the desired behavior of sigmoid function and adjusting its values in the structure boundaries. This takes less time-consuming that search in all CT images that include part of the structure.

6. REFERENCES

[1] D. Aykac, E. A. Hoffman, G. McLennan, and J. M. Reinhardt, “Segmentation and analysis of the human airway tree from three-dimensional X-ray CT images.,” *IEEE Trans. Med. Imaging*, vol. 22, no. 8, pp. 940–950, Aug. 2003.

[2] W. Park, E. Hoffman, and M. Sonka, “Segmentation of intrathoracic airway trees: a fuzzy logic approach.,” *IEEE Trans. Med. Imaging*, vol. 17, no. 4, pp. 489–97, Aug. 1998.

[3] G. Székely, A. Kelemen, C. Brechbühler, and G. Gerig, “Segmentation of 2-D and 3-D objects from MRI volume data using constrained elastic deformations of flexible Fourier contour and surface models.,” *Med. Image Anal.*, vol. 1, no. 1, pp. 19–34, Mar. 1996.

[4] S. Gorthi, V. Duay, N. Houhou, M. B. Cuadra, U. Schick, M. Becker, A. S. Allal, and J. Thiran, “Segmentation of Head and Neck Lymph Node Regions for Radiotherapy Planning Using Active Contour-Based Atlas Registration,” *IEEE J. Sel. Top. Signal Process.*, vol. 3, no. 1, pp. 135–147, 2009.

[5] R. Pohle and K. D. Toennies, “Segmentation of medical images using adaptive region growing,” *SPIE*, pp. 1337–1346, Jul. 2001.

[6] J. Acharya, S. Gadhiya, and K. Raviya, “Segmentation Techniques for Image Analysis: A Review,” *Int. J. Comput. Sci. Manag. Res.*, vol. 2, no. 1, pp. 1218–1221, 2013.

[7] G. K. Seerha and R. Kaur, “Review on Recent Image Segmentation Techniques,” *Int. J.*

Comput. Sci. Eng., vol. 5, no. 02, pp. 109–112, 2013.

[8] H. C. Van Assen, “3D Active Shape Modeling for Cardiac MR and CT Image Segmentation,” in *3D Active Shape Modeling for Cardiac MR and CT Image Segmentation*, 2006.

[9] H. C. van Assen, M. G. Danilouchkine, M. S. Dirksen, J. H. C. Reiber, and B. P. F. Lelieveldt, “A 3-D active shape model driven by fuzzy inference: application to cardiac CT and MR.,” *IEEE Trans. Inf. Technol. Biomed.*, vol. 12, no. 5, pp. 595–605, Sep. 2008.

[10] N. R. Pal and S. K. Pal, “A Review on Image Segmentation Techniques,” *Pattern Recognit.*, vol. 26, no. 9, pp. 1277–1294, 1993.

[11] Z. Ma, J. M. Tavares, and R. M. Natal Jorge, “A Review on the Current Segmentation Algorithms for Medical Images,” in *Proceedings of the First International Conference on Computer Imaging Theory and Applications (IMAGAPP 2009)*, 2009.

[12] S. K. Somasundaram and P. Alli, “A Review on Recent Research and Implementation Methodologies on Medical Image Segmentation,” *J. Comput. Sci.*, vol. 8, no. 1, pp. 170–174, 2012.

[13] T. Kitasaka, H. Yano, M. Feuerstein, and K. Mori, “Bronchial region extraction from 3D chest CT image by voxel classification based on local intensity structure,” in *Third International Workshop on Pulmonary Image Analysis*, 2010, pp. 21–29.

[14] Y. Hirano, R. Xu, R. Tachibana, and S. Kido, “A Method for Extracting Airway Trees by Using a Cavity Enhancement Filter,” in *Fourth International Workshop on Pulmonary Image Analysis*, 2011, no. 21103008, pp. 91–99.

[15] R. Venkatraman, R. Raman, B. Raman, R. B. Moss, G. D. Rubin, L. H. Mathers, and T. E. Robinson, “Fully automated system for three-dimensional bronchial morphology analysis using volumetric multidetector computed tomography of the chest.,” *J. Digit. Imaging*, vol. 19, no. 2, pp. 132–9, Jun. 2006.

[16] C. Bauer, “Segmentation of 3D Tubular Tree Structures in Medical Images,” Graz University of Technology, Austria, 2010.

[17] S. J. Park, J. H. Kim, J. M. Goo, K. G. Kim, and S. H. Lee, “A Virtual Bronchoscopy with Color-mapped Wall Thickness,” in *APAMI 2006 in conjunction with MIST 2006*, 2006, pp. 661–666.

[18] X. Zhou, T. Hayashi, T. Hara, H. Fujita, R. Yokoyama, T. Kiryu, and H. Hoshi, “Automatic segmentation and recognition of anatomical lung structures from high-resolution chest CT images.,” *Comput. Med. Imaging Graph.*, vol. 30, no. 5, pp. 299–313, Jul. 2006.

[19] S. Osher and J. A. Sethian, “Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations,” *J. Comput. Phys.*, vol. 79, pp. 12–49, 1988.

[20] J. A. Sethian, “Theory , Algorithms , and Applications of Level Set Methods for Propagating Interfaces,” *to Appear. Press. Acta Numer.*, 1995.

[21] J. A. Sethian, “Evolution, Implementation, and Application of Level Set and Fast Marching Methods for Advancing Fronts,” *J. Comput. Phys.*, vol. 169, no. 2, pp. 503–555, May 2001.

[22] E. W. Dijkstra, “A Note on Two Problems in Connexion with Graphs,” *Numer. Math.*, vol. 1, pp. 269–271, 1959.

[23] E. N. Marieb and K. Hoehn, *Human Anatomy & Physiology*, Eighth Edi. Pearson Learning Solutions, 2012.

[24] H. J. Johnson, M. M. Mccormick, and L. Ibañez, “The ITK Software Guide Book 1 : Introduction and Development Guidelines Fourth Edition Updated for ITK version 4 . 6,” 2014.

[25] J. F. Canny, “A Computational Approach to Edge Detection,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, 1986.

[26] R. Deriche, “Fast algorithms for low-level vision,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 1, pp. 78–87, 1990.

[27] J. F. Canny, “Finding Edges and Lines in Images,” Massachusetts Institute of Technology, 1983.

[28] R. Deriche, “Recursively Implementing the Gaussian and its Derivatives,” INRIA Sophia-Antipolis, 1993.

[29] G. Farnebäck and C.-F. Westin, “Improving Deriche-style Recursive Gaussian Filters,” *J. Math. Imaging Vis.*, vol. 26, no. 3, pp. 293–299, Nov. 2006.