2013 IEEE International Conference on Computer Vision

Accurate and Robust 3D Facial Capture Using a Single RGBD Camera

Yen-Lin Chen
Texas A&M University

Hsiang-Tao Wu
Microsoft Research Asia

Fuhao Shi
Texas A&M University

Xin Tong
Microsoft Research Asia

Jinxiang Chai
Texas A&M University

Abstract

This paper presents an automatic and robust approach that accurately captures high-quality 3D facial performances using a single RGBD camera. The key of our approach is to combine the power of automatic facial feature detection and image-based 3D nonrigid registration techniques for 3D facial reconstruction. In particular, we develop a robust and accurate image-based nonrigid registration algorithm that incrementally deforms a 3D template mesh model to best match observed depth image data and important facial features detected from single RGBD images. The whole process is fully automatic and robust because it is based on single frame facial registration framework. The system is flexible because it does not require any strong 3D facial priors such as blendshape models. We demonstrate the power of our approach by capturing a wide range of 3D facial expressions using a single RGBD camera and achieve state-of-the-art accuracy by comparing against alternative methods.

1. Introduction

The ability to accurately capture 3D facial performances has many applications including animation, gaming, human-computer interaction, security, and telepresence. This problem has been partially solved by commercially available marker-based motion capture equipment (*e.g.*, [18]), but this solution is far too expensive for common use. It is also cumbersome, requiring the user to wear more than 60 carefully positioned retro-reflective markers on the face. This paper presents an alternative to solving this problem: reconstructing the user's 3D facial performances using a single RGBD camera.

The main contribution of this paper is a novel 3D facial modeling process that accurately reconstructs 3D facial expression models from single RGBD images. We focus on single frame facial reconstruction because it ensures the process is fully automatic and does not suffer from drifting errors. At the core of our system lies a 3D facial deformation registration process that incrementally deforms a template face model to best match observed depth data. We model 3D facial deformation in a reduced subspace through embedded deformation [16] and extend model-based optical flow formulation to depth image data. This allows us to formulate the 3D nonrigid registration process in the Lucas-Kanade registration framework [1] and use linear system solvers to incrementally deform the template face model to match observed depth images.

Our image-based 3D nonrigid registration process, like any other iterative registration processess [1], requires a good initialization. The system often produces poor registration results when facial deformations are far from the template face model. In addition, it does not take into account perceptually significant facial features such as nose tip and mouth corners, thereby resulting in misalignments in those perceptually important facial regions. We address the challenges by complementing our image-based nonrigid registration process with automatic facial feature detection process. Our experiment shows that incorporating important facial features into the nonrigid registration process significantly improves the accuracy and robustness of the reconstruction process.

We demonstrate the power of our facial reconstruction system by modeling a wide range of facial expressions using a single *Kinect* (see Figure 1). We evaluate the performance of the system by comparing against alternative methods including marker-based motion capture [18], "faceshift" system [20], Microsoft face SDK [11], and nonrigid facial registration using Iterative Closest Points (ICP).

2. Background

Our system accurately captures high-quality 3D facial performances using a single RGBD camera. Therefore, we will focus our discussion on methods and systems developed for acquiring 3D facial performances.

One of most successful approaches for 3D facial performance capture is based on marker-based motion cap-

1550-5499/13 $31.00 © 2013 IEEE
DOI 10.1109/ICCV.2013.449

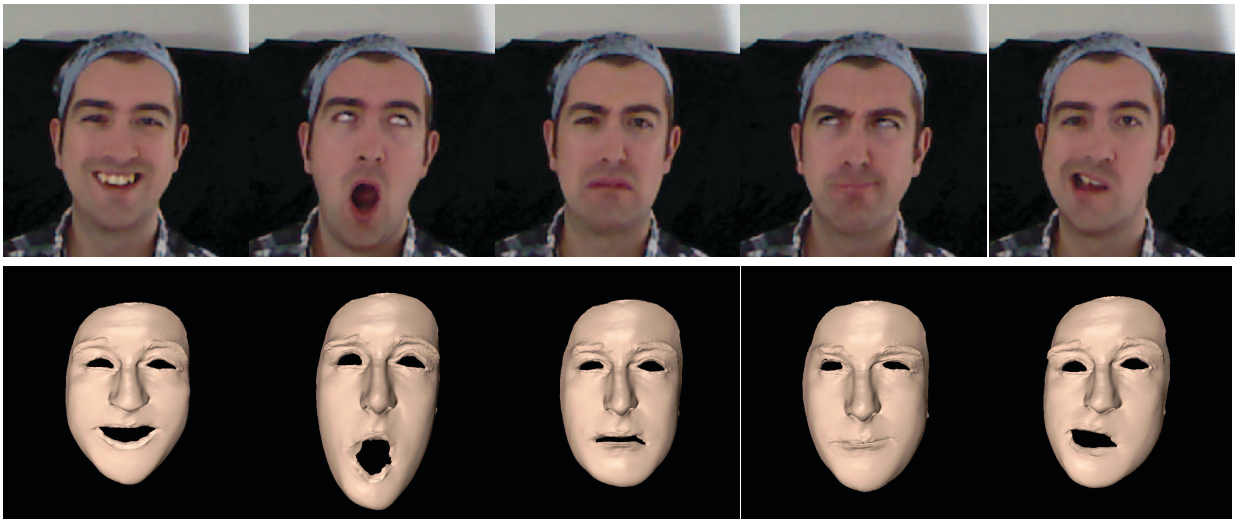

Figure 1. Accurate and robust facial performance capture using a single *Kinect*: (top) reference image data; (bottom) the reconstructed facial performances.

ture [9], which can robustly and accurately track a sparse set of markers attached on the face. Recent effort in this area (*e.g.*, [2, 10]) has been focused on complementing marker-based systems with other capturing types of devices such as video cameras and/or 3D scanners to improve the resolution and details of captured facial geometry. Marker-based motion capture, however, is not practical for random users targeted by this paper as they are expensive and cumbersome for 3D facial performance capture.

Marker-less motion capture provides an appealing alternative to facial performance capture because it is non-intrusive and does not impede the subject's ability to perform facial expressions. One solution to marker-less facial capture is the use of depth and/or color data obtained from structured light systems [22, 13, 12, 21]. For example, Zhang and his colleagues [22] captured 3D facial geometry and texture over time and built the correspondences across all the facial geometries by deforming a generic face template to fit the acquired depth data using optical flow computed from image sequences. Recently, Li and his colleagues [12] captured dynamic depth maps with their real-time structured light system and fit a smooth template to the captured depth maps.

The minimal requirement of a single camera for facial performance capture is particularly appealing, as it offers the lowest cost and a simplified setup. However, previous single RGB camera systems for facial capture [7, 6, 14] are often vulnerable to ambiguity caused by a lack of distinctive features on face and uncontrolled lighting environments. One way to address the issue is to use 3D prior models to reduce the ambiguity of image-based facial deformations (*e.g.*, [3, 19]). More recent research [5, 4, 20] has been focused on modeling 3D facial deformation using a single RGBD camera such as *Microsoft Kinect* or *time-of-flight (TOF)* cameras. For example, Cai and colleagues [5] explored how to use a linear deformable model constructed by an artist and Iterative Closest Points (ICP) techniques to fit deformable model from depth data. Breidt et al. [4] constructed 3D identity and expression morphable models from a large corpus of prerecorded 3D facial scans and used them to fit depth data obtained from a *ToF* camera via similar ICP techniques.

Among all the systems, our work is most closely related to Weise et al. [20], which uses RGBD image data captured by a single *Kinect* and a template, along with a set of predefined blend shape models, to track facial deformations over time. Our system shares a similar perspective as theirs because both are targeting low-cost and portable facial capture accessible to random users. Our goal, however, is different from theirs in that we focus on authentic reconstruction of 3D facial performances rather than performance-based facial retargeting and animation. Our method for facial capture is also significantly different from theirs. Their approach utilizes a set of predefined blend shape models and closest points measurement to sequentially track facial performances in a Maximum A Posteriori (MAP) framework. In contrast, our approach focuses on single frame facial reconstruction and combines image-based registration techniques with automatic facial detection in the Lucas-Kanade registration framework. Another difference is that we model deformation using embedded deformation rather than blendshape representation and therefore do not require any predefined blendshape models, which significantly reduces overhead costs for 3D facial capture. Lastly, as shown in our comparison experiment, our system achieves much more accurate results than Weise et al. [20].

(a) (b) (c) (d)

Figure 2. Accurate and robust facial performance capture using a single *Kinect*: (a) the input depth image; (b) the input color image; (c) the detected facial features superimposed on the color image; (d) the reconstructed 3D facial model using both the depth image and detected facial features.

3. Overview

Our system acquires high-quality 3D facial models from single RGBD images recorded by a single *Kinect*. A *Kinect* can simultaneously capture depth maps with a resolution of 320×240 and color images with a resolution of 640×480 at 30 frames per second based on infrared projection. Our facial modeling process leverages automatic facial feature detection and image-based nonrigid registration for 3D facial expression modeling (Figure 2).

We start the process by automatically locating important facial features such as the nose tip and the mouth corners in single RGBD images (Figure 2(c)). Briefly, we formulate feature detection as a per-pixel classification problem and apply randomized trees to associate each pixel with probability scores of being a particular feature. The detected features are often noisy and frequently corrupted by outliers due to classification errors. To handle this challenge, we employ geometric hashing to robustly search closest examples in a training set of labeled images, where all the key facial features are labeled, and remove misclassified features inconsistent with the closest example. In the final step, we refine feature locations by utilizing active appearance models (AAM) and 2D facial priors embedded in K closest examples of the detected features. For more details of our feature detection process, please refer to [15].

We model 3D facial deformation using embedded deformation [16] and this allows us to constrain the solution to be in a reduced subspace. We introduce a model-based depth flow algorithm to incrementally deform the face template to best match observed depth data as well as detected facial features. We formulate the problem in the Lucas-Kanade image registration framework and incrementally estimates both rigid transformations (ρ) and nonrigid deformation ($\mathbf{g}$) via linear system solvers. In addition, we introduce extra terms into the registration framework to further improve the robustness and accuracy of our system.

4. Image-based Nonrigid Facial Registration

This section focuses on automatic 3D facial modeling using single RGBD images, which is achieved by deforming a template mesh model, $\mathbf{s}_0$, to best match observed image data. In our implementation, we obtain the template mesh model $\mathbf{s}_0$ by scanning facial geometry of the subject under a neutral expression.

4.1. Facial Configuration Space

We model nonrigid deformation $\mathbf{g}$ using embedded deformation representation developed by Sumner et al. [16]. Embedded deformation builds a space deformation represented by a collection of affine transformations organized in a graph structure. One affine transformation is associated with each node and induces a deformation on the nearby space. The influence of nearby nodes is blended by the embedded deformation algorithm in order to deform the vertices or the graph nodes themselves. We choose embedded deformation because it allows us to model the deformation in a reduced subspace, thereby significantly reducing the ambiguity for 3D facial modeling.

In embedded deformation, the affine transformation for an individual node is defined by a 3-by-3 matrix $\mathbf{A}_j$ and a 3-by-1 translation vector $\mathbf{t}_j$. In this way, the collection of all per-node affine transformations, denoted as $\mathbf{g} = \{\mathbf{A}_j, \mathbf{t}_j\}_{j=1,...,M}$, where M is the total number of the nodes, expresses a non-rigid deformation of the template mesh model in a reduced deformation space. Specifically, the deformed position $\tilde{\mathbf{v}}_i$ of each shape vertex $\mathbf{v}_i$ is a weighted sum of its positions after application of the affine transformations associated with the k closest nodes to the vertex:

$$\tilde{\mathbf{v}}_i = \sum_{j=1}^{k} w_j(\mathbf{v}_i)[\mathbf{A}_j(\mathbf{v}_i - \mathbf{n}_j) + \mathbf{n}_j + \mathbf{t}_j] \qquad (1)$$

where $\{\mathbf{A}_j, \mathbf{t}_j\}_{j=1,...,k}$ are the affine transformations associated with the k closest nodes of the vertex $\mathbf{v}_i$ and $\mathbf{n}_j$ is the node position for the j-th node on the template mesh. The weights $w_j(\mathbf{v}_i), j = 1, ..., k$, are spatially varying and thus

depend on the vertex position. The weights for each vertex are precomputed in a way similar to [16]. In our experiment, graph nodes are chosen by uniformly sampling vertices of the template mesh model in the frontal facial region. We have found $M = 250$ graph nodes are often sufficient to model facial details captured by a *Kinect*. We also experimentally set k to 4.

We represent rigid transformations of the face using a 6-by-1 vector ρ, which stacks the translation and rotation vectors. The state of our facial modeling process can thus be defined by $\mathbf{q} = [\rho, \mathbf{g}]$. We denote the deformed mesh model as $\mathbf{s} = \mathbf{s}_0 \oplus \mathbf{q}$, where the operator $\oplus$ represents the application of both rigid transformations and embedded deformations to the template mesh $\mathbf{s}_0$.

Let $\bar{\mathbf{p}} = [\bar{x}, \bar{y}, \bar{z}]$ be the *barycentric* coordinates of a point on the surface of the template mesh $\mathbf{s}_0$. The global coordinates of the corresponding point on the surface of the deforming mesh $\mathbf{s}$ are defined by a forward kinematics function $\mathbf{h}$ for mesh deformation: $\mathbf{p} = \mathbf{h}(\mathbf{q}; \bar{\mathbf{p}}, \mathbf{s}_0)$, which computes the global position ($\mathbf{p}$) of a surface point from the model state ($\mathbf{q}$) given the local coordinates ($\mathbf{p}_0$) of a surface point on the template mesh ($\mathbf{s}_0$). We model the relationship between the global coordinates of a surface point and its corresponding pixel on image plane using a projection transformation function obtained from *Kinect SDK*: $\mathbf{x} = \mathbf{f}(\mathbf{p})$, where $\mathbf{x} = [u, v]$ represents the 2D coordinates of the corresponding pixel in depth image.

4.2. Objective Function

We adopt an "analysis-by-synthesis" strategy to measure how well the transformed and deformed face template model fits observed RGBD image data. Our image-based nonrigid facial registration process aims to minimize the following objective function:

$$\min_{\mathbf{q}} E_{data} + \alpha_1 E_{rot} + \alpha_2 E_{reg} \tag{2}$$

where the first term is the *data fitting* term, which measures how well the deformed template model matches the observed RGBD data. The second term E_{rot} ensures that local graph nodes deform as rigidly as possible (*i.e.*, $\mathbf{A}_j^T \mathbf{A}_j = \mathbf{I}_{3\times3}$). The third term E_{reg} serves as a regularizer for the deformation by indicating that the affine transformations of adjacent graph nodes should agree with one another. The weights α_1 and α_2 control the importance of the second and third terms. In our experiment, we set the weights α_1 and α_2 to 1.0 and 0.5, respectively. Here we focus our discussion on the first term. For details about the second and third term, please refer to the original work of embedded deformation [16].

We define the data fitting term as a weighted combination of three terms:

$$\alpha_{depth} E_{depth} + \alpha_{feature} E_{feature} + \alpha_{boundary} E_{boundary} \tag{3}$$

where the first term E_{depth} is the *depth image* term which minimizes the difference between the "observed" and the "hypothesized" depth data. The second term $E_{feature}$ is the *facial feature* term which ensures the "hypothesized" facial features are consistent with the "detected" facial features in observed data. The third term is the *boundary* term which stabilizes the registration process by penalizing the misalignments of boundary points between the "hypothesized" and "observed" face models. In our experiment, we set the weights $\alpha_{feature}$, α_{depth}, and $\alpha_{boundary}$ to 0.001, 0.1 and 5, respectively.

This requires minimizing a sum of squared nonlinear function values. Our idea is to extend the Lucas-Kanade algorithm [1] to solve the above non-linear least squares problem. Lucas-Kanade algorithm, which is a Gauss-Newton gradient descent non-linear optimization algorithm, assumes that a current estimate of $\mathbf{q}$ is known and then iteratively solves for increments to the parameters $\delta\mathbf{q}$ using linear system solvers.

4.2.1 Depth Image Term

This section introduces a novel model-based depth flow algorithm for incrementally estimating rigid transformation (ρ) and nonrigid transformation ($\mathbf{g}$) of the template mesh ($\mathbf{s}_0$) to best match the observed depth image D. Assume the movement ($\delta\mathbf{p}$) between the two frames to be small, the depth image constraint at $D(\mathbf{x}, t)$ is defined as follows:

$$D(\mathbf{x}(\mathbf{p}), t) + \delta z = D(\mathbf{x}(\mathbf{p} + \delta\mathbf{p}), t + 1). \tag{4}$$

Intuitively, when a 3D surface point ($\mathbf{p}$) has a delta movement ($\delta\mathbf{p}$) in 3D space, its projected pixel $\mathbf{x}(\mathbf{p})$ on image plane will have the corresponding movement $\delta\mathbf{x} = (\delta u, \delta v)$. However, unlike color image registration via optical flow, the depth value of a pixel is not constant. Instead, it will produce a corresponding small change δz along the depth axis. This is due to reparameterization of 3D point $\mathbf{p}$ on 2D image space.

Similar to the *optical flow* formulation, we derive the *depth flow* formulation by approximating the right side of Equation (4) with a Taylor series expansion. We have

$$(\frac{\partial D}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial \mathbf{p}} - \frac{\partial z}{\partial \mathbf{p}}) \delta\mathbf{p} + \frac{\partial D}{\partial t} = 0, \tag{5}$$

where partial derivatives $\partial D / \partial \mathbf{x}$ are gradients of the depth image at pixel $\mathbf{x}$. The derivatives $\partial \mathbf{x} / \partial \mathbf{p}$ can be evaluated by the projection function $\mathbf{x} = \mathbf{f}(\mathbf{p})$. The temporal derivative $\partial D / \partial t$ simply measures the pixel difference of two consecutive depth images. The partial derivatives $\frac{\partial z}{\partial \mathbf{p}}$ is simply a row vector $[0, 0, 1]$.

We adopt an "analysis-by-synthesis" strategy to incrementally register the deforming template mesh with the observed depth image via depth flow. More specifically, we

render a depth image based on the current model state ($\mathbf{q}$) and then estimate an optimal update of the model state ($\delta\mathbf{q}$) by minimizing the inconsistency between the "observed" and "rendered" depth images. To register the deforming template face with the observed depth image via *depth flow*, we associate the delta movement of a point ($\delta\mathbf{p}$) with the delta change of the model state ($\delta\mathbf{q}$):

$$\delta\mathbf{p} = \frac{\partial\mathbf{h}(\mathbf{q};\mathbf{p}_0,\mathbf{s}_0)}{\partial\mathbf{q}}\delta\mathbf{q}, \tag{6}$$

where the vector-valued function $\mathbf{h}$ is the forward kinematics function for mesh deformation [17].

After combing Equation (5) with Equation (6) using chain rules, we have

$$(\frac{\partial D}{\partial\mathbf{x}}\frac{\partial\mathbf{x}}{\partial\mathbf{p}} - \frac{\partial z}{\partial\mathbf{p}})\frac{\partial\mathbf{h}}{\partial\mathbf{q}}\delta\mathbf{q} + \frac{\partial D}{\partial t} = 0. \tag{7}$$

The above equation shows how to optimally update the model state ($\delta\mathbf{q}$) based on spatial and temporal derivatives of the "rendered" depth image $D(u,v)$. In the model-based depth flow formulation, we evaluate the spatial derivatives based on the gradients of the "rendered" depth image. The temporal derivatives $\frac{\partial D}{\partial t}$ are evaluated by the difference between the "observed" and "rendered" depth images. An optimal update of the model state can be achieved by summing over the contributions of individual depth pixels associated with the template face.

A remaining issue for the *depth image* term evaluation is to determine which pixels in the "rendered" depth image should be included for evaluation. We stabilize the model-based depth flow estimation process by excluding the pixels outside the border of outer boundary of the face. Corresponding vertices on the template mesh are automatically marked by back projecting the border pixels of the rendered depth image. Similarly, we remove the pixels that are inside the border of inner boundary of the face, in particular the mouth and eyes. The inner boundary of the face is defined by the close regions of the detected facial features of the mouth and eyes.

4.2.2 Facial Feature Term

Depth data alone is often not sufficient to model accurate facial deformation because it does not take into account perceptually significant facial features such as the nose tip and the mouth corners, thereby resulting in misalignments in those perceptually important facial regions. We address the challenge by including the *facial feature* term into the objective function. In our implementation, we choose to define the *facial feature* term based on a combination of 2D and 3D facial points obtained from detection process.

In preprocessing step, we annotate the locations of facial features on the template mesh model by identifying the barycentric coordinates ($\bar{\mathbf{p}}_i$) of facial features on the template mesh. The *facial feature* term minimizes the inconsistency between the "hypothesized" and "observed" features in either 2D or 3D space:

$$\sum_i \omega_i\|\mathbf{f}(\mathbf{h}(\mathbf{q};\bar{\mathbf{p}}_i,\mathbf{s}_0)) - \mathbf{x}_i\|^2 + (1-\omega_i)\|\mathbf{h}(\mathbf{q};\bar{\mathbf{p}}_i,\mathbf{s}_0) - \mathbf{p}_i\|^2$$

where the vector $\mathbf{x}_i$ and $\mathbf{p}_i$ are 2D and 3D coordinates of the i-th detected facial features. The weight ω_i is a binary value, which returns "1" if depth information is missing, otherwise "0". Note that only facial features around important regions, including the mouth and nose, eyes, and eyebrows, are included for *facial feature* term evaluation. This is because facial features located on outer contour are often not very stable.

4.2.3 Boundary Term

Depth data from a $Kinect$ is often very noisy and frequently contains missing data along the face boundary. This inevitably results in noisy geometry reconstruction around the face boundary. We introduce the *boundary* term to stabilize the registration along the boundary.

To handle noisy depth data around the outer boundary of the face, we first estimate the rigid-body transformation ρ that aligns the template mesh with observed data (see Section 4.3). During nonrigid registration process, we stabilize the outer boundary of the deforming face by penalizing the deviation from the transformed template $\mathbf{s}_0 \oplus \rho$. Vertices on the outer boundary of the template/deforming mesh are automatically marked by back projecting outer boundary pixels of the "rendered" depth image. We define the *boundary* term in 3D position space by minimizing the sum of the squares of the distances between the boundary vertices of the deforming mesh ($\mathbf{s}_0 \oplus \rho \oplus \mathbf{g}$) and their target 3D positions obtained from the transformed mesh ($\mathbf{s}_0 \oplus \rho$).

4.3. Registration Optimization

Our 3D facial modeling requires minimizing a sum of squared nonlinear function values defined in Equation (2). Our idea is to extend the Lucas-Kanade framework [1] to solve the non-linear least squares problem. Lucas-Kanade algorithm assumes that a current estimate of $\mathbf{q}$ is known and then iteratively solves for increments to the parameters $\delta\mathbf{q}$ using linear system solvers. In our implementation, we start with the template mesh and iteratively transform and deform the template mesh until the change of the state $\mathbf{q}$ is smaller than a specified threshold.

We have observed that a direct estimation of rigid transformations and embedded deformation is prone to local minima and often produces poor results. We thus decouple rigid transformations from nonrigid deformation and solve

them in two sequential steps. In the first step, we drop off the *boundary* term from the objective function defined in Equation (3) and estimate the rigid transformation ρ using iterative linear solvers. We stabilize the rigid alignment by using a pre-segmented template that excludes the chin region from the registration as this part of the face typically exhibits the strongest nonrigid deformations. In the second step, we keep the computed rigid transformation constant and iteratively estimate embedded deformation $\mathbf{g}$ based on the objective function defined in Equation (2).

This requires minimizing a sum of squared nonlinear function values. Our idea is to extend the Lucas-Kanade algorithm [1] to solve the above non-linear least squares problem using iterative linear system solvers. In our implementation, we analytically evaluate the Jacobian terms of the objective function. The fact that each step in the registration algorithm can be executed in parallel allows implementing a fast solver on modern graphics hardware.

5. Algorithm Evaluation

Our evaluation consists of two parts. The first part compares our approach with alternative methods. The second part validates the proposed approach by evaluating the importance of each key component of our process. We have evaluated the effectiveness of our algorithm based on both synthetic and real data. Our results are best seen in the accompanying video. We also show sample frames of the evaluation results in our supplementary PDF file.

Evaluation on synthetic data. We evaluate our system on synthetic RGBD image data generated by high-fidelity 3D facial data captured by Huang and colleagues [10]. The whole testing sequence consists of 1388 frames. We first synthesize a sequence of color and depth images based on a RGBD camera (*i.e.*, *Kinect*) setting similar to what occurs in the real world. The resolutions of image and depth data, therefore, are set to 640×480 and 320×240, respectively, with 24-bit RGB color values and 13-bit integer depth values in millimeters. The face models are placed at a distance to approximate real world capturing scenarios.

We test our algorithm as well as alternative methods on synthetic RGBD images and obtain the quantitative error of the algorithm by comparing its reconstruction data against ground truth data. In particular, we compute the average correspondence/reconstruction error between our reconstructed models and the ground truth data across the entire sequence. We evaluate the reconstruction error by measuring the sum of distances between vertex position on the reconstruction mesh and its corresponding position on the ground truth mesh. Note that we know the correspondences between the two meshes because the reconstruction meshes are deformed from the first mesh of ground truth data.

Evaluation on real data. We further evaluate the performance of our system on real data by comparing against ground truth facial data acquired by an optical motion capture system [18]. We placed 62 retro-reflective markers (4 mm diameter hemispheres) on the subject's face and set up a twelve-camera *Vicon* motion capture system [18] to record dynamic facial movements at 240 frames per second acquisition rate. We synchronize a *Kinect* camera with the optical motion capture system to record the corresponding RGBD image data. The whole test sequence consists of 838 frames corresponding a wide range of facial expressions. We test our algorithm as well as alternative methods on recorded RGBD image data. We use the 3D trajectories of 62 markers captured by *Vicon* as ground truth data to evaluate the performance of the algorithm. The captured marker positions and observed RGBD image data are from different coordinate systems and therefore we need to transform them into the same coordinate system. To achieve this goal, we use the video images from *Kinect* as a reference to manually label marker positions in the *Kinect* coordinate system and use them to estimate an optimal 4×4 transformation matrix to transform markers from the motion capture coordinate system to the *Kinect* coordinate system.

	Our Method	Nonrigid ICP
x (2D image plane)	0.64997 pixel	2.5890 pixel
y (2D image plane)	0.85933 pixel	4.0202 pixel
x (3D space)	1.2112 mm	4.7829 mm
y (3D space)	1.5817 mm	7.5349 mm
z (3D space)	2.1866 mm	4.8383 mm

Table 1. Quantitative evaluation of our algorithm and Nonrigid ICP registration using ground truth data scaptured with a full marker set in a twelve-camera *Vicon* system [18].

5.1. Comparisons Against Alternative Methods

We have evaluated the performance of our system by comparing against alternative methods.

Comparison against *Vicon* system. In this experiment, we quantitatively assess the quality of the captured motion by comparing with ground truth motion data captured with a full marker set in a twelve-camera *Vicon* system [18]. The average reconstruction error, which is computed as average 3D marker position discrepancy between the reconstructed facial models and the ground truth mocap data, was reported in Table 1. The accompanying video also shows a side-by-side comparison between our result and the result obtained by *Vicon*. The computed quantitative errors provide us an upper bound on the actual errors because of reconstruction errors from the *Vicon* mocap system and imperfect alignments of the *Vicon* markers with the *Kinect* data. Note that we attached the mocap markers on the subject's face, which makes the markers deviate from the actual surface of the face.

Comparison against nonrigid ICP techniques. We

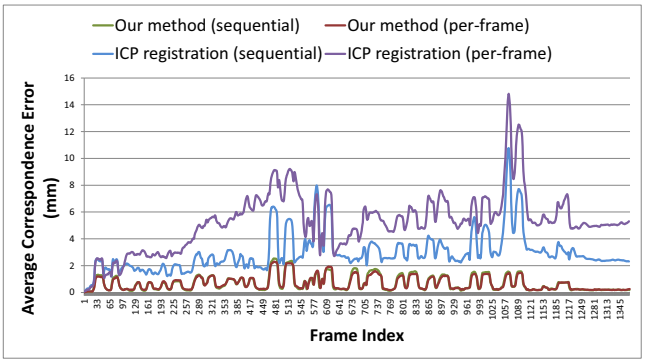

Figure 3. Average correspondence/reconstruction errors across the entire sequence for nonrigid ICP registration and our method with/without temporal coherence. The evaluation is based on synthetic RGBD image data from [10].

compare our method against non-rigid ICP method on ground truth data obtained from the *Vicon* system. The nonrigid ICP process is similar to [12]. Both our method and nonrigid ICP are built on embedded deformation with the same settings. The only difference is the way to define the data term (*i.e.*, E_{data} in Equation (2)). Nonrigid ICP process estimates both rigid transformations and embedded deformation with standard iterative closest point techniques (ICP). In each iteration, the corresponding point of each vertex on the deforming mesh is found by its closest point on the observed depth data. The corresponding points of all the vertices are then used to update the deforming mesh. Both methods estimate a rigid transformation, followed by the non-rigid deformation. The results shown in Table 1 as well as the accompanying video show that our system produces much better results than non-rigid ICP. This is because ICP is often sensitive to initial values and prone to local minimum, particularly involving tracking high-dimensional facial mesh model from noisy depth data.

In addition, we have compared our system against nonrigid ICP registration on synthetic data generated by high-quality facial performance data obtained by [10]. We compare them in two different ways: single frame registration and sequential tracking. Sequential tracking incorporates temporal coherence into facial reconstruction process. More specifically, we include a smoothness term into the objective function to penalize the change of the reconstructed meshes in two consecutive frames. In addition, we utilize the result from previous frame to initialize the current frame. As shown in Figure 3, our facial reconstruction produces more accurate results over nonrigid ICP registration, with and without temporal coherence.

Comparison against Weise et al. [20]. We compare our results against Weise et al. [20]. We downloaded their tracking software "faceshift" [8]. We started their tracking process by building a personal profile using their system. Specifically, we instructed the user to sit in front of a *Kinect* and recorded a small number of facial expressions to retarget a set of predefined blendshape models to the user [20]. With the retargeted blendshape models, we can use their software to sequentially track the facial expression of the user. The accompanying video clearly shows our system produces much more accurate results than their system.

Comparison against Microsoft Kinect Facial SDK [11]. We evaluate the performance of our system by doing a side-by-side comparison against Microsoft Kinect facial SDK. Kinect facial SDK tracks a number of facial features (about 100 features) from RGBD image data and combine the tracked facial features with a small number of predefined blendshape models in the Candide3 model [5], including six AUs (Animation Units) and 11 SUs (Shape Units), to reconstruct facial deformation. The accompanying video shows our system produces much better results. This is because we model detailed deformation of the whole face using both facial features and per-pixel depth information.

5.2. Evaluation on 3D Facial Reconstruction Process

We have evaluated the performance of our facial registration process by dropping off each term of the cost function described in Equation 3. The evaluation is based on synthetic RGBD image data.

The importance of facial feature term. We compared results obtained by the facial registration process with or without the *facial feature* term. The accompanying video shows that tracking without the *facial feature* term often results in misalignments of perceptually important facial features. More importantly, when the facial performances are very different or far away from the template model, the optimization often gets stuck in local minima, thereby producing inaccurate results. With the *facial feature* term, the algorithm can accurately reconstruct facial performances across the entire sequence. Without the *facial feature* term, the average correspondence/reconstruction error is increased from 0.68 mm per vertex to 1.71 mm per vertex.

The importance of depth image term. Our evaluation shows that incorporating the *depth image* term into the objective function can significantly improve the reconstruction accuracy. This is because the *facial feature* term only constrains facial deformation at locations of a sparse set of facial features rather than detailed per-pixel constraints. With the *depth image* term, the average error of our facial reconstruction process is reduced from 5.46 mm per vertex to 0.68 mm per vertex.

The importance of boundary term. We evaluate the importance of the *boundary* term on both synthetic data and real data. The accompanying video shows that adding the *boundary* term stabilizes the facial deformation along the

border of face boundary, which is more obvious in the real data.

5.3. Applications in Facial Performance Capture

We have tested our system on acquiring 3D facial performances of four subjects. We used a Minolta VIVID 910 laser scanner to record high-resolution static facial geometry of an actor/actress as the template mesh. Our results are best seen in the accompanying video. In our implementation, we utilize the temporal coherence to speed up the facial tracking process. The algorithm usually converges quickly as we initialize the solution using the result from the previous frames. The current system runs at an interactive frame rate (2 frames per second) on a machine with Intel Core i7 3.40GHz CPU and GeForce GTX 580 graphics card.

6. Conclusion

We have presented an automatic algorithm for accurately capture 3D facial performances using a single RGBD camera. The key idea of our algorithm is to combine the power of image-based 3D nonrigid registration and automatic facial feature detection for 3D facial registration. We demonstrate the power of our approach by modeling a wide range of 3D facial expressions using a single RGBD camera and achieve state-of-the-art accuracy by comparing against alternative methods.

Our system is appealing for 3D facial modeling and capture. It is flexible because it does not require strong 3D facial priors commonly used in previous facial modeling systems (*e.g.*, [20]). It is robust and does not suffer from error accumulation because it builds on single frame facial registration framework. Last but not the least, our system is accurate because facial feature detection provides locations for significant facial features to avoid misalignments and our image-based nonrigid registration method achieves down to sub-pixel accuracy.

References

[1] S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 2004. 56(3):221–255. 1, 4, 5, 6

[2] B. Bickel, M. Botsch, R. Angst, W. Matusik, M. Otaduy, H. Pfister, and M. Gross. Multi-scale capture of facial geometry and motion. *ACM Trans. Graph.*, 26(3):33:1–33:10, 2007. 2

[3] V. Blanz, C. Basso, T. Poggio, and T. Vetter. Reanimating faces in images and video. In *Computer Graphics Forum*, 2003. 22(3):641–650. 2

[4] M. Breidt, H. Biilthoff, and C. Curio. Robust semantic analysis by synthesis of 3d facial motion. In *Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, pages 713–719. IEEE, 2011. 2

[5] Q. Cai, D. Gallup, C. Zhang, and Z. Zhang. 3d deformable face tracking with a commodity depth camera. pages 229–242, 2010. 2, 7

[6] D. DeCarlo and D. Metaxas. Optical flow constraints on deformable models with applications to face tracking. *International Journal of Computer Vision*, 2000. 38(2):99-127. 2

[7] I. Essa, S. Basu, T. Darrell, and A. Pentland. Modeling, tracking and interactive animation of faces and heads using input from video. In *Proceedings of Computer Animation Conference*. 1996. 68-79. 2

[8] faceshift. http://www.faceshift.com/, 2013. 7

[9] B. Guenter, C. Grimm, D. Wood, H. Malvar, and F. Pighin. Making Faces. In *Proceedings of ACM SIGGRAPH 1998*, pages 55–66, 1998. 2

[10] H. Huang, J. Chai, X. Tong, and H.-T. Wu. Leveraging motion capture and 3d scanning for high-fidelity facial performance acquisition. *ACM Trans. Graph.*, 30(4):74:1–74:10, 2011. 2, 6, 7

[11] Kinect for Windows SDK. http://msdn.microsoft.com/en-us/library/jj130970.aspx/, 2013. 1, 7

[12] H. Li, B. Adams, L. J. Guibas, and M. Pauly. Robust single-view geometry and motion reconstruction. *ACM Trans. Graph.*, 28(5):175:1–175:10, 2009. 2, 7

[13] W.-C. Ma, A. Jones, J.-Y. Chiang, T. Hawkins, S. Frederiksen, P. Peers, M. Vukovic, M. Ouhyoung, and P. Debevec. Facial performance synthesis using deformation-driven polynomial displacement maps. *ACM Trans. Graph.*, 27(5):121:1–121:10, 2008. 2

[14] F. Pighin, R. Szeliski, and D. Salesin. Resynthesizing facial animation through 3D model-based tracking. In *International Conference on Computer Vision*. 1999. 143–150. 2

[15] F. Shi and J. Chai. Fast and accurate facial alignment from a single rgbd image. In *Computer Science and Engineering Technical Reports 2013, Texas A&M University*. 2013. 3

[16] R. W. Sumner, J. Schmid, and M. Pauly. Embedded deformation for shape manipulation. *ACM Trans. Graph.*, 26(3):80:1–80:7, 2007. 1, 3, 4

[17] R. W. Sumner, M. Zwicker, C. Gotsman, and J. Popović. Mesh-based inverse kinematics. *ACM Trans. Graph.*, 24(3):488–495, July 2005. 5

[18] Vicon Systems. http://www.vicon.com, 2013. 1, 6

[19] D. Vlasic, M. Brand, H. Pfister, and J. Popović. Face transfer with multilinear models. *ACM Transactions on Graphics*, 24(3):426–433, 2005. 2

[20] T. Weise, S. Bouaziz, H. Li, and M. Pauly. Realtime performance-based facial animation. *ACM Trans. Graph.*, 30(4):77:1–77:10, 2011. 1, 2, 7, 8

[21] T. Weise, H. Li, L. Van Gool, and M. Pauly. Face/off: live facial puppetry. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '09, pages 7–16, New York, NY, USA, 2009. ACM. 2

[22] L. Zhang, N. Snavely, B. Curless, and S. Seitz. Spacetime faces: high resolution capture for modeling and animation. *ACM Transactions on Graphics*, 23(3):548–558, 2004. 2