Visual Comput (2005) 21: 629–638
DOI 10.1007/s00371-005-0346-7

ORIGINAL ARTICLE

Scalable 3D video of dynamic scenes

Michael Waschbüsch
Stephan Würmlin
Daniel Cotting
Filip Sadlo
Markus Gross

Published online: 31 August 2005

M. Waschbüsch (✉) · S. Würmlin · D. Cotting · F. Sadlo · M. Gross
Computer Graphics Laboratory
Department of Computer Science
Swiss Federal Institute of Technology (ETH), Zurich
Switzerland
{waschbuesch, wuermlin, dcotting, sadlof, grossm}@inf.ethz.ch

Abstract In this paper we present a scalable 3D video framework for capturing and rendering dynamic scenes. The acquisition system is based on multiple sparsely placed 3D video bricks, each comprising a projector, two grayscale cameras, and a color camera. Relying on structured light with complementary patterns, texture images and pattern-augmented views of the scene are acquired simultaneously by time-multiplexed projections and synchronized camera exposures. Using space–time stereo on the acquired pattern images, high-quality depth maps are extracted, whose corresponding surface samples are merged into a view-independent, point-based 3D data structure. This representation allows for effective photo-consistency enforcement and outlier removal, leading to a significant decrease of visual artifacts and a high resulting rendering quality using EWA volume splatting. Our framework and its view-independent representation allow for simple and straightforward editing of 3D video. In order to demonstrate its flexibility, we show compositing techniques and spatiotemporal effects.

Keywords 3D video · Free-viewpoint video · Scene acquisition · Point-based graphics

1 Introduction

As one of the many promising emerging technologies for home entertainment and spatiotemporal visual effects, 3D video acquires the dynamics and motion of a scene during recording while providing the user with the possibility to change the viewpoint at will during playback. Free navigation regarding time and space in streams of visual data directly enhances the viewing experience and interactivity. Unfortunately, in most existing systems virtual viewpoint effects have to be planned precisely and changes are no more feasible after the scene has been shot. As an example, Digital Air's *Movia*® systems comprise high-speed, high-definition digital cinema cameras that are placed accurately such that no software view interpolation is needed. But as a consequence, postprocessing and editing possibilities are restricted.

A number of multiview video systems allow for realistic rerenderings of 3D video from arbitrary novel viewpoints. However, for producing high-quality results, the capturing systems are confined to configurations where cameras are placed very close together. As an example, Zitnick et al. [38] covered a horizontal field of view of 30° with 8 cameras, where only linear arrangements were possible. For configurations that cover an entire hemisphere with a small number of cameras, either model-based approaches need to be employed (e.g., Carranza et al. [6] with 8 cameras) or degradation in visual quality has to be accepted (e.g., Würmlin et al. [33] with 16 cameras). The latter two systems are also limited by the employed reconstruction algorithms to the capture of foreground objects or even humans only, and scalability in terms of camera configurations and data structures is not addressed. Moreover, the underlying representations and processes typically do not allow for convenient editing.

Our work is motivated by the drawbacks of the aforementioned systems and by the vision of bringing 3D video to a new level where not only capturing and subsequent high-quality rerendering is cost-effective, convenient, and scalable, but also editing of the spatiotemporal streams is easy to perform. We envision 3D video editing to become as convenient as 2D home video editing. For this purpose, we rely on view-independent 3D geometry streams, which allow for similar authoring and editing techniques as carried out in common 3D content creation and modeling tools. Inserting novel objects to a scene or adding spatiotemporal effects is becoming straightforward with simple postprocessing methods, and one no longer has to cope with the common limitations of image-based representations.

Specifically, we make the following contributions in this paper:

- We introduce sparsely placed, scalable 3D video bricks that act as low-cost z-cameras and allow simultaneous texture and depth map acquisition using space–time stereo on structured light.
- We propose a view-independent point-based representation of the depth information acquired by the 3D video bricks. Different postprocessing algorithms enable image generation from novel viewpoints with appealing quality.
- We present a probabilistic rendering technique based on view-dependent EWA volume splatting, providing clean images by smoothly blending noise from the reconstruction process.
- We demonstrate the suitability of our view-independent 3D representation for authoring and editing and show several results of 3D video, including effects like object cloning and motion trails.

2 Related work

This paper extends or integrates previous work in areas like point-based computer graphics, depth-from-stereo, and 3D video. For the sake of conciseness, we refer the reader to the ACM SIGGRAPH 2004 course on point-based computer graphics [1] and to relevant depth-from-stereo publications [27, 37]. In the following discussion, we will confine ourselves to related work in the area of 3D video.

In 3D video, multiview video streams are used to rerender a time-varying scene from arbitrary viewpoints. There is a continuum of representations and algorithms suited for different acquisition setups and applications. Purely image-based representations [18] need many densely spaced cameras for applications like 3D-TV [21]. Dynamic light field cameras [32, 35], which have camera baselines of a couple of centimeters, do not need any geometry at all. Camera configuration constraints can be relaxed by adding more and more geometry to image-based systems, as demonstrated by Lumigraphs [10]. Voxel-based representations [30] can easily integrate information from multiple cameras but are limited in resolution. Depth-image-based representations [2, 28] use depth maps that are computed predominantly by stereo algorithms [9, 35, 38]. Stereo systems still require reasonably small baselines and, hence, scalability and flexibility in terms of camera configurations is still not achieved. Redert et al. [26] use depth images acquired by Zcams [13] for 3D video broadcast applications. Appropriate representations for coding 3D audio/visual data are currently investigated by the MPEG-4 committee [29]. On the other end of the continuum are model-based representations that describe the objects or the scene by time-varying 3D geometry, possibly with additional video textures [6, 14]. Almost arbitrary camera configurations become feasible, but most existing systems are restricted to foreground objects only.

Besides data representations, one has to distinguish between online and offline applications. Matusik et al. [19, 20] focus on real-time applications, e.g., 3D video conferencing or instant 3D replays. However, they are restricted to capturing foreground objects only due to the nature of their silhouette-based depth reconstruction algorithms. Gross et al. [11] use a 3D video system based on a point sample representation [33] for their telecollaboration system *blue-c* and share the same limitation of only being able to reconstruct foreground objects. Mulligan et al. [22] also target telepresence. They compute geometric models with multicamera stereo and transmit texture and depth over a network. Carranza et al. [6] present an offline 3D video system that employs an a priori shape model that is adapted to the observed outline of a human. However, this system is only able to capture predefined shapes, i.e., humans. The 3D video recorder [34] handles point-sampled 3D video data captured by silhouette-based reconstruction algorithms and discusses data storage issues. No full scene acquisition is possible with the last two systems, but almost arbitrary camera configurations are possible. Zitnick et al. [38] proposed a layered depth image representation for high-quality video view interpolation. Reconstruction errors at depth discontinuities are smoothed out by Bayesian matting. However, this approach again needs a quite dense camera setup to generate high-quality renderings in a limited viewing range. Scalability to larger setups is not addressed by the authors.

Cockshott et al. [7] also propose a 3D video studio based on modular acquisition units and pattern-assisted stereo. For concurrent texture acquisition, the patterns are projected using strobe lights requiring custom-built hardware. Only foreground objects are modeled using implicit surfaces.

3 Overview

Our 3D video acquisition system consists of several so-called 3D video bricks that capture high-quality depth maps from their respective viewpoints using calibrated pairs of stereo cameras (Fig. 1). The matching algorithm used for depth extraction is assisted by projectors illuminating the scene with binary structured light patterns. Alternating projection of a pattern and its inverse allows for concurrent acquisition of the scene texture using appropriately synchronized color cameras.

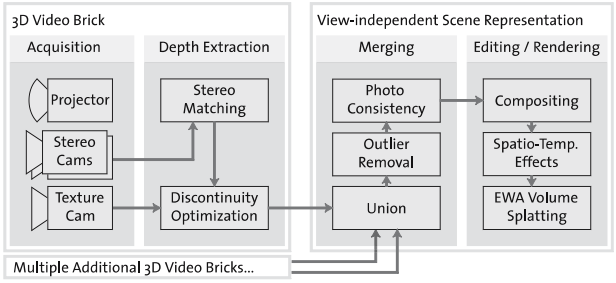

Fig. 1. Overview of 3D video framework

The depth maps are postprocessed to optimize discontinuities, and the results from different viewpoints are unified into a view-independent, point-based scene representation consisting of Gaussian ellipsoids. During merging, we remove outliers by ensuring photoconsistency of the point cloud with all acquired images from the texture cameras. Editing operations like compositing and spatiotemporal effects can now be applied to the view-independent geometry. Novel viewpoints of the dynamic scene are rendered using EWA volume splatting.

4 Scalable 3D video bricks

In this section we present the concept of our low-cost z-cameras realized by 3D video bricks allowing simultaneous acquisition of textures and depth maps.

4.1 Acquisition setup

The basic building blocks of the 3D video setup are movable bricks containing three cameras and a projector illuminating the scene with alternating patterns. Two grayscale cameras are responsible for depth extraction, while a color camera acquires the texture information of the scene. Figure 2 shows a single brick prototype with its components. In our current implementation, we operate with three bricks, each consisting of a standard PC with a genlock graphics board (NVIDIA Quadro FX3000G), a projector synchronizing to the input signal (NEC LT240K), and cameras having XGA resolution

Fig. 2. 3D video brick with cameras and projector (*left*), simultaneously acquiring textures (*middle*), and structured light patterns (*right*)

(Point Grey Dragonfly). The components are mounted on a portable aluminum rig as shown in Fig. 2. The system is complemented by a synchronization microcontroller (MCU) connected to the cameras and the genlock-capable graphics boards.

At a certain point in time, each brick can only capture depth information from a particular fixed position. In order to span a wider range of viewpoints and reduce occlusion effects, multiple movable bricks can be combined and individually oriented to cover the desired working space as illustrated in Fig. 3). Scalability of multiple bricks is guaranteed because overlapping projections are explicitly allowed by our depth reconstruction and because the computation load of each brick does not increase during real-time recording. Each brick performs the grabbing completely independently of the other bricks with the exception of the frames being timestamped consistently by using a common synchronization device.

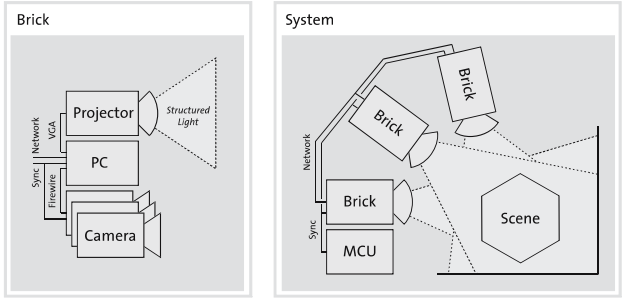

Fig. 3. Configuration of our 3D video prototype system

In order to compute valid depth maps and merge the information gained from several bricks, all cameras in the 3D video system must be calibrated intrinsically and extrinsically. We determine imaging properties of all cameras using the MATLAB camera calibration toolbox [3]. The projectors do not need to be calibrated.

4.2 Simultaneous texture and depth acquisition

Each brick concurrently acquires texture information with the color camera and depth information using the stereo

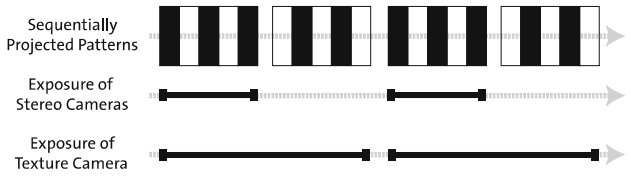

Fig. 4. Camera exposure with inverse pattern projection

pair of grayscale cameras. Stereovision (Sect. 4.3) generally requires a highly texturized scene to find good correlations between different views. It generally fails in reconstructing simple geometry of uniformly colored objects, e.g., white walls. Additionally, the textures should be nonperiodic to guarantee unique matches. As a consequence, we add artificial textures to the scene by projecting structured light patterns, as originally proposed by Kang et al. [16]. We use a binary vertical stripe pattern with randomly varying stripe widths. It supports strong and unique correlations in the horizontal direction and is at the same time insensitive to vertical deviations that may occur from inaccuracies in the camera calibration. To avoid untexturized shadows, the scene is illuminated by patterns from all bricks at the same time. Our approach has the advantage of being insensitive to interferences between the different projections and does not need a projector calibration, unlike pure structured light approaches or stereo matching between cameras and projectors.

Alternating projections of structured light patterns, and the corresponding inverses allow for simultaneous acquisition of the scene textures using an appropriately synchronized texture camera as illustrated in Fig. 4. Note that this camera does not see the patterns emanating from the projector, but only a constant white light, which preserves the original scene texture (Fig. 2).

Since the patterns are changing at a limited rate of 60 Hz (projector input frequency), flickering is slightly visible to the human eye. Alternative solutions using imperceptible structured light [8] do not show any flickering, but require faster, more sensitive, and, therefore, more expensive cameras for reliable stereo depth extraction.

4.3 Stereo matching on structured light

Each brick acquires the scene geometry using a depth-from-stereo algorithm. Depth maps are computed for the images of the left and right grayscale cameras by searching for corresponding pixels. To reduce occlusion problems between the views, the cameras are mounted at a small horizontal baseline of 20 cm.

In recent decades, a large variety of stereo algorithms has been developed. A survey of different methods and their implementations can be found in Scharstein et al. [27]. Recently, Zitnick et al. [38] used a segmentation-based algorithm to generate high-quality 3D video. Experiments have shown that the approach works well for conventional passive stereo but fails on structured light patterns, where segments seem to become too small and too similar to result in unique correlations. According to the authors, their method is more suited for multibaseline stereo applications.

Our work is based on space–time stereo [36, 37] that exploits time coherence to correlate the stereo images and computes disparities with subpixel accuracy. The authors formulate stereo matching as a maximization problem over an energy $E(d, d_u, d_v, d_t)$ that defines a matching criterion of two pixel correlation windows. The solution delivers both the disparity d and its derivatives d_u, d_v in image space and d_t in the time domain. In contrast to the original work, we employ the maximum normalized cross correlation (MNCC) as similarity measure, which is robust against global color variations between the left and right image. In our implementation, we optimize E using the downhill-simplex method. To prevent the algorithm from converging to a local minimum, we use the disparities of neighboring pixels as a starting guess for the optimization and additionally repeat the process with several random starting values. We consider the maximum value of E as a measure of the correlation quality and reject all disparities with an energy below a certain threshold.

The number of potential outliers can be decreased by extending the correlation window in the temporal dimension to cover three or more images. However, because correlation assumes continuous surfaces, there arise some artifacts at depth discontinuities. For moving scenes, discontinuities in the image space are extended into the temporal domain, making correlation computation even more difficult. For complex dynamic scenes, we therefore use an adaptive correlation window covering multiple time steps only in static parts of the images that can be detected by comparing successive frames. Remaining errors are smoothed out with our discontinuity optimization approach presented in the next section.

4.4 Discontinuity optimization

We smooth discontinuity artifacts by applying a two-phase postprocessing to the disparity images. First, we conservatively identify the regions of wrong disparities. Second, we extrapolate new disparities into these regions from their neighborhoods.

In the first phase, the detection of occluded parts is performed by a simple cross checking. We perform the stereo correlation twice, between the left and right image and vice versa. Corresponding pixels in both disparity maps should have the same values, otherwise they belong to occlusions. This way we mask out all pixels whose disparities differ about more than one pixel.

The second phase operates on depth images computed from the disparity maps. We detect remaining outliers and

fill all missing depths by extrapolating from their neighbors. To recover the correct discontinuities we block the extrapolation at texture edges of the color images. As the color textures are acquired by another camera, we first warp the depth map into the texture camera. Then, we decompose the texture image into segments of similar colors using simple color quantization. With a high probability, all pixels in one color segment belong to one continuous surface and therefore have similar depths. Differing depths can be considered as outliers, which are identified by clustering the depths in each color segment: the biggest cluster is assumed to represent the correct surface; pixels of all smaller clusters are removed. The holes in the depth map are then filled by a moving least squares extrapolation [17] constrained by the current color segment.

Figure 5 shows a comparison of the original depth image with an image computed by our discontinuity optimization algorithm. For illustration we embedded the edges of the texture images. Notice how holes are filled and the depth discontinuities tightly fit to the color edges. Nevertheless, errors in the color segmentation can still lead to some outliers at discontinuities. However, most of them are eliminated during view merging as discussed in Sect. 5.2.

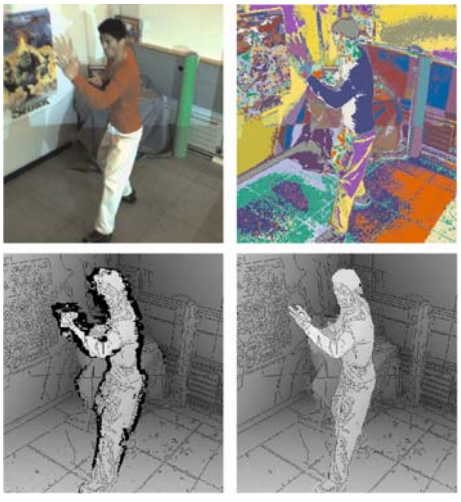

Fig. 5. Discontinuity optimization. *Upper left*: color image, *upper right*: color segmentation. Comparison of original depth image (*lower left*) with the one after our discontinuity optimization (*lower right*). For illustration, color edges have been embedded

5 View-independent scene representation

To model the resulting 3D scene, we propose a view-independent, point-based data representation. By merging all reconstructed views into a common world reference frame, we achieve a convenient, scalable representation: additional views can be added very easily by back-projecting their image pixels. Our model is in principle capable of providing a full 360° view if the scene has been acquired from enough viewpoints. Unlike image-based structures, it is possible to keep the amount of data low by removing redundant points from the geometry [24]. Compared to mesh-based methods, points provide advantages in terms of scene complexity because they reduce the representation to the absolutely necessary data and do not carry any topological information, which is often difficult to acquire and maintain. As each point in our model has its own assigned color, we also do not have to deal with texturing issues. Moreover, our view-independent representation is very suitable for 3D video-editing applications since tasks like object selection or relighting can be achieved easily with standard point-processing methods [1].

5.1 Point-based data model

Our point-based model consists of an irregular set of samples, where each sample corresponds to a point on a surface and describes its properties such as location and color. The samples can be considered as a generalization of conventional 2D image pixels toward 3D video. If required, the samples can be easily extended with additional attributes like surface normals for relighting.

To avoid artifacts in rerendering, we have to ensure full surface coverage of the samples. Thus, our samples cannot be represented by infinitesimal points but need to be considered as small surface or volume elements. One obvious representation are surfels [25], which are small elliptical disks aligned tangentially to the surface. However, surfels do not handle noise due to inaccurate 3D reconstruction or camera calibration very well and require accurate geometries and therefore stable surface normals.

Therefore, we have chosen a different approach, similar to that of Hofsetz et al. [12]. Every point is modeled by a 3D Gaussian ellipsoid spanned by the vectors $\boldsymbol{t}_1$, $\boldsymbol{t}_2$, and $\boldsymbol{t}_3$ around its center $\boldsymbol{p}$. This corresponds to a probabilistic model describing the positional uncertainty of each point by a trivariate normal distribution

$$p_X(\boldsymbol{x}) = N(\boldsymbol{x};\, \boldsymbol{p},\, \boldsymbol{V}) = \frac{1}{\sqrt{(2\pi)^3 |\boldsymbol{V}|}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{p})^{\mathrm{T}} \boldsymbol{V}^{-1} (\boldsymbol{x}-\boldsymbol{p})}\,, \tag{1}$$

with expectation value $\boldsymbol{p}$ and covariance matrix

$$\boldsymbol{V} = \Sigma^{\mathrm{T}} \cdot \Sigma = \begin{pmatrix} \boldsymbol{t}_1 & \boldsymbol{t}_2 & \boldsymbol{t}_3 \end{pmatrix}^{\mathrm{T}} \cdot \begin{pmatrix} \boldsymbol{t}_1 & \boldsymbol{t}_2 & \boldsymbol{t}_3\,, \end{pmatrix} \tag{2}$$

composed of 3×1 column vectors $\boldsymbol{t}_i$.

To estimate $\boldsymbol{V}$, Hofsetz et al. [12] have chosen an approach based on the quality of the pixel correlation of the stereo matching. It turns out that these resulting heuristic uncertainties are quite large compared to the high-quality

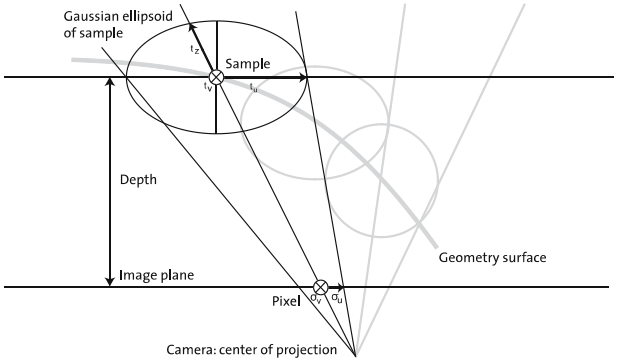

Fig. 6. Construction of a 3D Gaussian ellipsoid

disparities we are able to obtain from our structured-light-assisted approach. Consequently, we propose a different approach that constrains the uncertainties to cover only small but well-defined acquisition errors. We assume that most disparities are correctly estimated up to small errors caused by deviations in the camera calibration and compute point sizes that just provide full surface coverage.

Assuming a Gaussian model for each image pixel uncertainty, we first compute the back-projection of the pixel into three-space, which is a 2D Gaussian parallel to the image plane spanned by two vectors $\boldsymbol{t}_u$ and $\boldsymbol{t}_v$. Extrusion into the third domain by adding a vector $\boldsymbol{t}_z$ guarantees a full surface coverage under all possible views. This is illustrated in Fig. 6.

Each pixel (u, v) is spanned by orthogonal vectors $\sigma_u(1, 0)^{\mathrm{T}}$ and $\sigma_v(0, 1)^{\mathrm{T}}$ in the image plane. Assuming a positional deviation σ_c, the pixel width and height under uncertainty are $\sigma_u = \sigma_v = 1 + \sigma_c$. σ_c is estimated to be the average reprojection error of our calibration routine.

The depth z of each pixel is inversely proportional to its disparity d as defined by the equation

$$z = -\frac{f_L \cdot \|\boldsymbol{c}_L - \boldsymbol{c}_R\|}{d + p_L - p_R}\,, \tag{3}$$

where f_L is the focal length of the left rectified camera, $\boldsymbol{c}_L$ and $\boldsymbol{c}_R$ are the centers of projection, and p_L and p_R the u-coordinates of the principal points. The depth uncertainty σ_z is obtained by differentiating Eq. 3 and augmenting the gradient Δd of the disparity with its uncertainty σ_c:

$$\sigma_z = \frac{f_L \cdot \|\boldsymbol{c}_L - \boldsymbol{c}_R\|}{(d + p_L - p_R)^2} \cdot (\Delta d + \sigma_c)\,. \tag{4}$$

Now, we can construct for each pixel its Gaussian in ray space with

$$\Sigma_R = \begin{pmatrix} \sigma_u \cdot z & 0 & \sigma_z \cdot u \\ 0 & \sigma_v \cdot z & \sigma_z \cdot v \\ 0 & 0 & \sigma_z \end{pmatrix}. \tag{5}$$

This is transformed into the world coordinate system by

$$\Sigma = \boldsymbol{P}^{-1} \cdot \Sigma_R \tag{6}$$

using the camera projection matrix $\boldsymbol{P}$.

The centers $\boldsymbol{p}$ of the ellipsoids are constructed by back-projection as

$$\boldsymbol{p} = \boldsymbol{P}^{-1} \cdot (u, v, 1)^{\mathrm{T}} \cdot z + \boldsymbol{c}\,, \tag{7}$$

where $\boldsymbol{c}$ is the center of projection of the camera.

5.2 Photoconsistency enforcement

After back-projection, the point model still contains outliers and falsely projected samples. Some points originating from a specific view may look wrong from extrapolated views due to reconstruction errors, especially at depth discontinuities. In the 3D model, they may cover correct points reconstructed from other views, disturbing the overall appearance of the 3D video. Thus, we remove those points by checking the whole model for photoconsistency with all texture cameras.

After selecting a specific texture camera, we successively project each ellipsoid (as computed in Sect. 5.1) into the camera image in increasing depth order, starting with the points closest to the camera. We determine all pixels of the original image that are covered by the projection and not yet occluded by previously tested, valid ellipsoids. We compare the average color of those pixels with the color of the ellipsoid. If both colors differ too much, the point sample is removed. Otherwise, we rasterize the ellipsoid into a z-buffer that is used for occlusion tests for all subsequent points.

As a result, enforcing photoconsistency considerably improves the seamless fit of multiple acquired depth maps in our model. The reduction of artifacts can be clearly seen in Fig. 7. Nevertheless, there remain some issues with mixed pixels, i.e., silhouette pixels possessing a color interpolated from different surfaces. These tend to produce holes in the cleaned model. This may be solved using boundary-matting techniques [15]. Currently, we apply our consistency check conservatively and tolerate remaining outliers which are not detected.

Fig. 7. Enforcing photo consistency during view merging: Without (*left*) and with (*right*) enforcement

6 Rendering

We render novel viewpoints of the scene using the GPU and CPU cooperatively. Smooth images are generated using the uncertainties of the Gaussian ellipsoids. Our method combines the advantages of two probabilistic image generation approaches described in Broadhurst et al. [4]. Additionally we perform a view-dependent blending similar to Hofsetz et al. [12].

6.1 Probabilistic rendering

Broadhurst et al. [4] use probabilistic volume ray casting to generate smooth images. Each ray is intersected with the Gaussians of the scene model. At a specific intersection point $\boldsymbol{x}$ with sample i, the evaluation $N(\boldsymbol{x}; \boldsymbol{p}_i; \boldsymbol{V}_i)$ of the Gaussian describes the probability that a ray will hit the corresponding surface point. To compute the final pixel color, two different approaches are described. The maximum likelihood method associates a color with the ray using only the sample with the most probable intersection. The second approach employs the Bayes rule: It integrates all colors along each ray weighted by the probabilities without considering occlusions. Thus, the color of a ray R is computed as

$$\boldsymbol{c}_R = \frac{\int_{\boldsymbol{x}\in R} \sum_i \boldsymbol{c}_i N(\boldsymbol{x}; \boldsymbol{p}_i, \boldsymbol{V}_i)}{\int_{\boldsymbol{x}\in R} \sum_i N(\boldsymbol{x}; \boldsymbol{p}_i, \boldsymbol{V}_i)} \,. \tag{8}$$

The maximum likelihood method generates crisp images, but it also sharply renders noise in the geometry. The Bayesian approach produces very smooth images with fewer noise but is incapable of handling occlusions and rendering solid surfaces in an opaque way.

We propose a rendering method that combines both approaches in order to benefit from their respective advantages. Our idea is to accumulate the colors along each ray as in the Bayesian setting but to stop as soon as a maximum accumulated probability has been reached. Reasonably, a Gaussian sample should be completely opaque if the ray passes its center. The line integral through the center of a 3D Gaussian has a value of $\frac{1}{2\pi}$ and for any ray R it holds that

$$\int_{\boldsymbol{x}\in R} N(\boldsymbol{x}; \boldsymbol{p}, \boldsymbol{V}) \leq \frac{1}{2\pi} \,. \tag{9}$$

Thus, we accumulate the solution of the integrals of Eq. 8 by traversing along the ray from the camera into the scene and stop as soon as the denominator of Eq. 8 reaches $\frac{1}{2\pi}$. Assuming that solid surfaces are densely sampled, the probabilities within the surface boundaries will be high enough so that the rays will stop within the front surface.

Fig. 8. Comparison of maximum likelihood (*left*) and Bayesian rendering (*center*) with our approach (*right*)

We compare the maximum likelihood and Bayesian rendering with our approach on noisy data in Fig. 8. Notice the large distortions in the maximum likelihood image that get smoothed out by the other two methods. However, the Bayesian renderer blends all the points including those from occluded surfaces, while our method renders opaque surfaces and maintains the blending. Thus, our renderer provides the advantages of both previous methods.

In our implementation, we replace the ray caster by a volume splatter [39] running on graphics hardware. After presorting the Gaussians according to their depths by the CPU, the GPU splats them from front to back. The pixel colors are blended according to the Gaussian alpha masks until the accumulated alphas reach a level of saturation. This is directly supported by the OpenGL blending function GL_SRC_ALPHA_SATURATE.

6.2 View-dependent blending

One specific sample usually looks most accurate from the view it has been acquired from. As the angle between the acquisition and the virtual view becomes larger, the quality decreases depending on the depth uncertainty of the Gaussian. Projections of samples with high uncertainty become more and more stretched, introducing visible artifacts, while samples with low uncertainties look good from all views. We treat this issue by applying the view-dependent blending of Hofsetz et al. [12]. The authors compute an alpha value representing the maximum opacity of each Gaussian in its center using the view-dependent criteria of Buehler et al. [5] weighted by the individual depth uncertainty σ_z.

Compared to a conventional surfel-based approach our combination of blending methods is able to better smooth out geometry noise in the model. This is clearly visible in Fig. 9.

7 Results and discussion

For the results presented in this section, we have recorded a dynamic scene with our setup consisting of three sparsely placed bricks covering an overall viewing angle of 70° horizontally and 30° vertically. Figure 10 shows

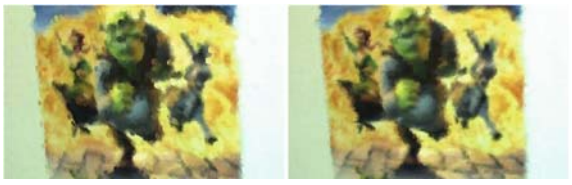

Fig. 9. Rendering using surfels (*left*) and our view-dependent uncertainty blending (*right*)

novel views of the acquired scene in Fig. 2, rendered from our reconstructed 3D model.

Our rerenderings have a decent look with a high-quality texture. Acquisition noise is smoothed out by our blending method. We are even able to reconstruct highly detailed geometry like the folds in the tablecloth shown in Fig. 11. However, there are still some artifacts at silhouettes that we would like to eliminate in the future. This is possible by using matting approaches as done by Zitnick et al. [38]. Some remaining outliers are also visible in the images. They could be reduced using a combination of multiple outlier removal algorithms [31] and by enforcing time coherence in the whole reconstruction pipeline. Big clusters of outliers tend to grow in our discontinuity optimization stage if they dominate the correct depths in a color segment. We are investigating hierarchical segmentation approaches as a possible solution. Furthermore, enforcing time coherence may help in filling remaining holes caused by occlusions.

Fig. 10. Re-renderings of the 3D video from novel viewpoints

Fig. 11. Geometric detail in the tablecloth. For illustration we recomputed smooth surface normals and rendered the scene with Phong lighting under two different illumination conditions

With our system we are able to acquire a large viewing range with a relatively low amount of cameras. To support increasingly large ranges, our system is scalable up to full spherical views. To fully cover 360° in all dimensions about 8 to 10 3D video bricks are needed. Note that this is not constrained to convex views. Although overlaps in the geometry can help to improve the overall quality due to the photoconsistency enforcement, they are not required as each brick reconstructs its own scene part independently.

The use of projectors still imposes some practical constraints because of the visible light spots and shadows that are created in the scene. We accept this limitation for the sake of a maximum 3D reconstruction quality. Using calibrated projectors it would be possible to compute the incident light at each surface point and compensate for the artifacts.

Our view-independent data model provides possibilities for novel effects and 3D video editing. Due to its point-based structure we are able to employ any kind of available point processing algorithms [1]. Once the 3D information is available, selection and compositing issues become straightforward and can be easily implemented using spatial clustering or bounding box algorithms. Such tasks are much harder to achieve on both conventional 2D video and view-dependent 3D video approaches based on light fields or depth maps only. Apart from the well-known time freeze we show two example effects in Fig. 12. We clone the actor by copying its corresponding point cloud to other places in the scene. Motion trails are generated by compositing semitransparent renderings of moving objects from previous timesteps.

Fig. 12. Special effects: actor cloning (*left*), motion trails (*right*)

8 Conclusions and future work

We presented a system for recording, processing, and rerendering 3D video of dynamic scenes. We are able to obtain high-quality depth maps using space–time stereo on structured light while concurrently acquiring textures of the scene. The brick concept combined with a view-independent data model allows for scalable capturing of a large viewing range with sparsely placed components. Decent-quality images of novel views are achieved using Gaussian ellipsoid rendering with view-dependent blending methods. Our point-based, view-independent data representation is well suited for spatiotemporal video editing. The representation can directly benefit from a large variety of available point-based processing algorithms, e.g., normal estimation [23] for relighting effects or simplification [24] for further redundancy elimination or level-of-detail rendering.

In the future, we will further investigate the editing capabilities of our representation. The ultimate goal is to provide a tool for spatiotemporal editing that is as easy to use as today's 2D video software and at the same time provides all novel possibilities that arise from a time-varying 3D scene representation. Furthermore, we would like to improve the resulting image quality by exploiting inter-brick correlation and eliminating remaining artifacts at silhouettes using matting approaches. It would also be desirable to achieve a comparable rendering quality using passive reconstruction algorithms only, which would make our system suitable for large outdoor scenes. In addition, we want to address compression of time-varying point-sampled 3D video streams.

Acknowledgement We would like to thank Stefan Rondinelli for implementing the stereo algorithm and Tim Weyrich for the fruitful discussions. This work is carried out in the context of the blue-c-II project, funded by ETH Grant No. 0-21020-04 as an internal polyproject.

References

1. Alexa, M., Gross, M., Pauly, M., Pfister, H., Stamminger, M., Zwicker, M.: Point-Based Computer Graphics. SIGGRAPH '04 Course Notes (2004)
2. Bayakovski, Y., Levkovich-Maslyuk, L., Ignatenko, A., Konushin, A., Timasov, D., Zhirkov, A., Han, M., Park, I.K.: Depth image-based representations for static and animated 3D objects. In: ICIP '02, **3**, 25–28 (2002)
3. Bouguet, J.Y.: Camera calibration toolbox for matlab, `http://www.vision.caltech.edu/bouguetj/calib_doc`
4. Broadhurst, A., Drummond, T., Cipolla, R.: A probabilistic framework for the space carving algorithm. In: ICCV '01, pp. 388–393 (2001)
5. Buehler, C., Bosse, M., McMillan, L., Gortler, S., Cohen, M.: Unstructured lumigraph rendering. In: SIGGRAPH '01, pp. 425–432 (2001)
6. Carranza, J., Theobalt, C., Magnor, M., Seidel, H.P.: Free-viewpoint video of human actors. In: SIGGRAPH '03, pp. 569–577 (2003)
7. Cockshott, W.P., Hoff, S., Nebel, J.C.: An experimental 3D digital TV studio. In: Vision, Image & Signal Processing '03, pp. 28–33 (2003)
8. Cotting, D., Naef, M., Gross, M., Fuchs, H.: Embedding imperceptible patterns into projected images for simultaneous acquisition and display. In: ISMAR '04, pp. 100–109 (2004)
9. Goldlücke, B., Magnor, M., Wilburn, B.: Hardware-accelerated dynamic light field rendering. In: VMV '02, pp. 455–462 (2002)
10. Gortler, S.J., Grzeszczuk, R., Szeliski, R., Cohen, M.F.: The lumigraph. In: SIGGRAPH '96, pp. 43–54 (1996)
11. Gross, M., Würmlin, S., Näf, M., Lamboray, E., Spagno, C., Kunz, A., Moere, A.V., Strehlke, K., Lang, S., Svoboda, T., Koller-Meier, E., Gool, L.V., Staadt, O.: blue-c: A spatially immersive display and 3D video portal for telepresence. In: SIGGRAPH '03, pp. 819–827 (2003)
12. Hofsetz, C., Ng, K., Max, N., Chen, G., Liu, Y., McGuinness, P.: Image-based rendering of range data with estimated depth uncertainty. IEEE CG&A **24**(4), 34–42 (2005)
13. Iddan, G.J., Yahav, G.: 3D imaging in the studio (and elsewhere ...). In: SPIE '01, **4298**, 48–55 (2001)
14. Kanade, T., Rander, P., Narayanan, P.J.: Virtualized reality: construction of virtual worlds from real scenes. IEEE Multimedia **4**(1), 34–47 (1997)
15. Kang, S., Szeliski, R.: Boundary matting for view synthesis. In: CVPRW '04 (2004)
16. Kang, S., Webb, J., Zitnick, L., Kanade, T.: A multi-baseline stereo system with active illumination and real-time image acquisition. In: ICCV '95, pp. 88–93 (1995)
17. Levin, D.: Mesh-independent surface interpolation. In: Geometric Modeling for Scientific Visualization, pp. 37–49, ed. by Brunnett, Hamann, Mueller, Springer 2003
18. Levoy, M., Hanrahan, P.: Light field rendering. In: SIGGRAPH '96, pp. 31–42 (1996)
19. Matusik, W., Buehler, C., McMillan, L.: Polyhedral visual hulls for real-time rendering. In: EGRW '01, pp. 115–125 (2001)
20. Matusik, W., Buehler, C., Raskar, R., Gortler, S.J., McMillan, L.: Image-based visual hulls. In: SIGGRAPH '00, pp. 369–374 (2000)
21. Matusik, W., Pfister, H.: 3D TV: A scalable system for real-time acquisition, transmission, and autostereoscopic display of dynamic scenes. In: SIGGRAPH '04 (2004)
22. Mulligan, J., Daniilidis, K.: View-independent scene acquisition for tele-presence. In: International Symposium on Augmented Reality, pp. 105–110 (2000)
23. Pauly, M., Gross, M.: Spectral processing of point sampled geometry. In: SIGGRAPH '01(2001)
24. Pauly, M., Gross, M., Kobbelt, L.: Efficient simplification of point-sampled geometry. In: VIS '02, pp. 163–170 (2002)
25. Pfister, H., Zwicker, M., van Baar, J., Gross, M.: Surfels: Surface elements as rendering primitives. In: SIGGRAPH '00, pp. 335–342 (2000)
26. Redert, A., de Beeck, M.O., Fehn, C., Ijsselsteijn, W., Pollefeys, M., Gool, L.V., Ofek, E., Sexton, I., Surman, P.: ATTEST: Advanced three-dimensional television system technologies. In: 3DPVT '02, pp. 313–319 (2002)
27. Scharstein, D., Szeliski, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. Int. J. Comput. Vis. **47**(1–3), 7–42 (2002)
28. Shade, J., Gortler, S., He, L.W., Szeliski, R.: Layered depth images. In: SIGGRAPH '98, pp. 231–242 (1998)
29. Smolic, A., Kimata, H.: Description of exploration experiments in 3DAV. In: JTC1/SC29/WG11 N6194. ISO/IEC (2003)
30. Vedula, S., Baker, S., Kanade, T.: Spatio-temporal view interpolation. In: EGRW '02, pp. 65–76 (2002)
31. Weyrich, T., Pauly, M., Keiser, R., Heinzle, S., Scandella, S., Gross, M.: Post-processing of scanned 3D surface

data. In: Eurographics Symposium on Point-Based Graphics '04 (2004)
32. Wilburn, B., Joshi, N., Vaish, V., Talvala, E.V., Antunez, E., Barth, A., Adams, A., Horowitz, M., Levoy, M.: High performance imaging using large camera arrays. In: SIGGRAPH '05 pp. 765–776 (2005)
33. Würmlin, S., Lamboray, E., Gross, M.: 3D video fragments: Dynamic point samples for real-time free-viewpoint video. In: Computers and Graphics '04 **28**(1), 3–14 (2004)
34. Würmlin, S., Lamboray, E., Staadt, O.G., Gross, M.H.: 3D video recorder. In: Proceedings of Pacific Graphics 2002, pp. 325–334. IEEE Press, New York (2002)
35. Yang, J.C., Everett, M., Buehler, C., McMillan, L.: A real-time distributed light field camera. In: EGRW '02, pp. 77–86 (2002)
36. Zhang, L., Curless, B., Seitz, S.M.: Spacetime stereo: shape recovery for dynamic scenes. In: CVPR '03, pp. 367–374 (2003)
37. Zhang, L., Snavely, N., Curless, B., Seitz, S.M.: Spacetime faces: high resolution capture for modeling and animation. In: SIGGRAPH '04, pp. 548–558 (2004)
38. Zitnick, C.L., Kang, S.B., Uyttendaele, M., Winder, S., Szeliski, R.: High-quality video view interpolation using a layered representation. In: SIGGRAPH '04, pp. 600–608 (2004)
39. Zwicker, M., Pfister, H., van Baar, J., Gross, M.: EWA splatting. IEEE Trans. Visual. Comput. Graph. **8**(3), 223–238 (2002)





MICHAEL WASCHBÜSCH is currently a Ph.D. candidate in the Computer Graphics Laboratory at ETH Zurich. In 2003, he received his computer science diploma degree from the University of Kaiserslautern, Germany. His research interests include 3D video, 3D reconstruction, point-based rendering, and graphics hardware.

STEPHAN WÜRMLIN is currently a postdoctoral researcher in the Computer Graphics Laboratory and project leader of the blue-c-II project (`http://blue-c-II.ethz.ch`). He received his Ph.D. from ETH Zurich in 2004 on the design of the 3D video technology for the blue-c collaborative virtual reality system. His current research interests include free-viewpoint video, point-based representations and rendering, real-time rendering, virtual reality, and multimedia coding.

DANIEL COTTING received his computer science diploma degree from ETH Zurich. He is currently enrolled in a Ph.D. program at the ETH Computer Graphics Laboratory led by Prof. Dr. Markus Gross. Daniel's current research interests include projection and display technologies, imperceptible structured light, augmented reality, interaction, and 3D reconstruction.

FILIP SADLO is a Ph.D. candidate in computer science at the Computer Graphics Laboratory of ETH Zurich, where he received his diploma in 2003. His research interests include scientific visualization, 3D reconstruction, and imaging.

MARKUS GROSS is a professor of computer science and director of the Computer Graphics Laboratory of ETH Zurich. He received a Master of Science in electrical and computer engineering and a Ph.D. in computer graphics and image analysis, both from the University of Saarbrucken, Germany. From 1990 to 1994 Dr. Gross worked for the Computer Graphics Center in Darmstadt, where he established and directed the Visual Computing Group. His research interests include point-based graphics, physics-based modeling, multiresolution analysis, and virtual reality. He has been widely publishing and lecturing on computer graphics and scientific visualization, and he authored the book *Visual Computing*, Springer, 1994. Dr. Gross has taught courses at major graphics conferences including ACM SIGGRAPH, IEEE Visualization, and Eurographics. He is the associate editor of IEEE Computer Graphics and Applications and has served as a member of international program committees of many graphics conferences. Dr. Gross has been a papers cochair of the IEEE Visualization '99, Eurographics 2000, and IEEE Visualization 2002 conferences. He is currently chair of the papers committee of ACM SIGGRAPH 2005.