FREE VIEWPOINT VIDEO EXTRACTION, REPRESENTATION, CODING, AND RENDERING

A. Smolic, K. Mueller, P. Merkle, T. Rein, M. Kautzner, P. Eisert, and T. Wiegand

Fraunhofer Institute for Telecommunications, Heinrich-Hertz-Institut
Image Processing Department
Einsteinufer 37, 10587 Berlin, Germany
{smolic/kmueller/merkle/rein/kautzner/eisert/wiegand}@hhi.de

ABSTRACT

Free viewpoint video provides the possibility to freely navigate within dynamic real world video scenes by choosing arbitrary viewpoints and view directions. So far, related work only considered free viewpoint video extraction, representation, and rendering methods. Compression and transmission has not yet been studied in detail and combined with the other components into one complete system. In this paper, we present such a complete system for efficient free viewpoint video extraction, representation, coding, and interactive rendering. Data representation is based on 3D mesh models and view-dependent texture mapping using video textures. The geometry extraction is based on a shape-from-silhouette algorithm. The resulting voxel models are converted into 3D meshes that are coded using MPEG-4 SNHC tools. The corresponding video textures are coded using an H.264/AVC codec. Our algorithms for view-dependent texture mapping have been adopted as an extension of MPEG-4 AFX. The presented results illustrate that based on the proposed methods a complete transmission system for efficient free viewpoint video can be built.

1. INTRODUCTION

Interactivity is an important key feature of audio-visual media applications. One type of interactivity is the ability to look around within a video scene by freely choosing viewpoints and viewing directions. The first media representations that provided such functionality were based on textured 3D mesh models and are well known from computer graphics, computer games and virtual reality applications.

Most of the scenes are either purely computer generated or contain static 2D views of real world objects represented by still pictures or moving textures. Recent achievements in image acquisition, processing, representation, and rendering also allow the user to navigate within dynamic real world scenes. Multiple cameras or special systems like omni-directional cameras are used to capture real world dynamic scenes. The acquired imagery is converted into a representation format that allows interactive rendering, i.e. selection of arbitrary viewpoints and/or directions. We refer to this type of data as free viewpoint video.

Suitable scene representation formats for free viewpoint video are often classified as a continuum between two extremes [1]. One extreme is the classical 3D computer graphics representation, where the scene geometry is typically described on the basis of 3D wire-frames and meshes. Real world objects are reproduced using geometric 3D surfaces with an associated texture being mapped onto them. More sophisticated attributes can be assigned as well. For instance, appearance properties (opacity, reflectance, light sources, etc.) can enhance the realism of the models significantly.

The other extreme is given by scene representations that do not use any 3D geometry at all, which is usually called image-based rendering. In this case, virtual intermediate views are generated from available real views by interpolation. The main advantages of image-based methods are a high quality of virtual view synthesis and an avoidance of a complex 3D scene reconstruction. However, these benefits are accompanied by the need for dense sampling of the real world with many original view images producing very large amounts of data.

In between the two extremes of purely geometry-based and fully image-based representations there exists a number of methods that make more or less use of both approaches and combine the advantages of both in a particular manner. In this paper, we present such a format for representation of dynamic real world 3D objects and scenes. It uses conventional 3D mesh models and view-dependent mapping of multiple textures as acquired from real cameras. The 3D geometry is reconstructed using a shape-from-silhouette algorithm that calculates a voxel approximation of the visual hull [2]. The voxel model is converted into a 3D mesh using a marching cubes algorithm and mesh reduction [3].

So far, related work only considered scene representation, construction, and rendering. Compression and transmission has not yet been studied in detail. In our paper, the geometry is coded using 3D mesh coding as standardized in MPEG-4 SNHC [4]. The multiple video textures are compressed using H.264/AVC [5]. For view-dependent texture mapping with unstructured Lumigraph rendering [6] we developed a new tool that was accepted as extension to MPEG-4 AFX [7].

This paper is organized as follows. In the next section, we describe the process of geometry extraction from multiple video streams. Section 3 presents the view-dependent texture mapping. Coding and rendering are described in section 4. Section 5 presents experimental results.

2. GEOMETRY RECONSTRUCTION

Free viewpoint video construction typically relies on a multi-camera setup as shown in Fig. 1. In general, the quality of the rendered views increases with the number of available cameras. However, equipment costs and often the complexity costs required for processing increase as well. We therefore consider a classical tradeoff between quality and costs by limiting the number of cameras and compensating this by geometry extraction.

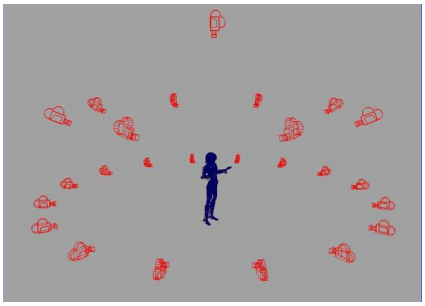

Fig. 1: Free viewpoint video acquisition

The first step of our algorithm consists of deriving intrinsic and extrinsic parameters for all cameras that relate the 2D images to a 3D world coordinate system as our geometry extraction and rendering algorithms require knowledge of these parameters. These parameters are computed from reference points using a standard calibration algorithm [8].

In the next step, the object to be extracted is segmented in all camera views. For that we use the combination of an adaptive background subtraction algorithm and Kalman filter tracking. The results of this step are silhouette videos that indicate the object's contour for all cameras. For details please refer to [9].

The 3D volume containing the object is reconstructed from the silhouette images using an octree-based shape-from-silhouette algorithm [10]. The process starts by placing a cube into the virtual 3D world that ideally represents the 3D bounding cube of the object. The size and position of the cube are obtained by projecting the 2D bounding boxes of all views into 3D space and analyzing the resulting intersections in 3D space. This initial cube of the octree, which is referred as level 0, is subdivided into 8 octants each of which being a cube itself. For each octant one of the following actions is taken:

1. A cube that is completely inside the silhouettes of all views is not subdivided further, i.e. it is completely inside the object to be reconstructed,
2. A cube that is completely outside of at least one silhouette is omitted, i.e. it is outside the object,
3. A cube that does not fall into one of the above two categories is further subdivided.

This approach is recursively applied to all octants, until a contour approximation with a particular accuracy in each view is achieved. Illustratively, the method of shape-from-silhouette can be compared to a carving process or the way a sculptor builds a figure from a block of marble. After visual hull segmentation the object's surface is extracted from the voxel model by applying a marching cubes algorithm [3].

3. VIEW-DEPENDENT TEXTURE MAPPING

For photo-realistic rendering, the original videos are mapped onto the reconstructed geometry. Natural materials may appear very different from different viewing directions depending on their reflectance properties and the lighting conditions. Static texturing (e.g. interpolating the available views) therefore often leads to poor rendering results, the so-called "painted shoebox" effect [11]. We have therefore developed a view-dependent texture mapping that more closely approximates natural appearance when navigating through the scene.

As illustrated in Fig. 2, the textures are projected onto the geometry using the calibration information. For each projected texture a normal vector $\boldsymbol{n}_i$ is defined pointing into the direction of the original camera. For generation of a virtual view into a certain direction $\boldsymbol{v}_{VIEW}$ a weight is calculated for each texture, which depends on the angle between $\boldsymbol{v}_{VIEW}$ and $\boldsymbol{n}_i$. The weighted interpolation ensures that at original camera positions the virtual view is exactly the original view. The closer the virtual viewpoint is to an original camera position the greater the influence of the corresponding texture on the virtual view.

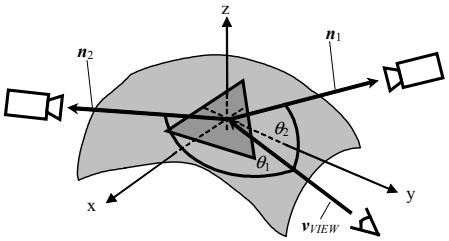

Fig. 2: Weighted virtual view interpolation

Fig. 3 illustrates the calculation of weights. θ_i are the angles between the virtual viewing direction and the camera directions $\boldsymbol{n}_i$. First individual weights w_i are calculated as:

$$w_i = \begin{cases} \dfrac{\cos\theta_i}{1-\cos\theta_i}, & \cos\theta_i \neq 1 \\ Float\_Max, & \cos\theta_i = 1 \end{cases}. \quad (1)$$

If one angle approaches zero, the weight approaches infinity. All other weights can be neglected in this case. In order to avoid overlighting the weights are normalized such that they always sum up to one:

$$a_i = \frac{w_i}{\sum_{\forall i} w_i}. \quad (2)$$

This weighted interpolation approximates realistic intermediate views enabling realistic virtual camera flights through the scene.

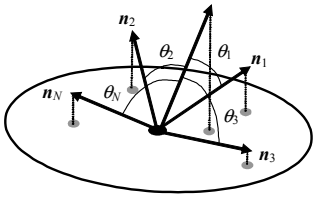

Fig. 3: Calculation of weights [12]

4. REPRESENTATION, RENDERING, AND CODING WITH MPEG-4 AND H.264/AVC

As mentioned earlier, previous work on free viewpoint video was mainly concentrated on extraction and rendering. Coding and transmission issues have not been considered so far. To investigate these issues, ISO/IEC MPEG has established a working group called 3DAV [13]. MPEG-4 is a suitable framework for free viewpoint video since it already provides state-of-the-art components for computer graphics and video representation and coding. However, view-dependent texture mapping as described in the previous section is not supported so far. We therefore did propose the technology as a new part of the Animation Framework eXtension (AFX) of MPEG-4. This part of the standard specifies advanced computer graphics tools [14]. Our proposal for view-dependent texture mapping was adopted for an upcoming amendment since the technology is not only suitable for free viewpoint video but for other computer graphics applications as well.

For coding of the 3D meshes, MPEG-4 already provides efficient tools [4] that can readily be used for our free viewpoint video representation. H.264/AVC [5], which is a joint standard of ITU-T VCEG and MPEG, is used for coding of the dynamic textures that are in fact video sequences.

Additionally, the camera calibration parameters have to be transmitted once to the decoder as side information, which means 10 float values per camera and session. A suitable coding format is under investigation in 3DAV.

5. EXPERIMENTS

A problem for evaluation of free viewpoint video is that for virtual intermediate views there are no original views to compare with, which is a fundamental difference to classical 2D video coding. With a large enough test set including many cameras it is possible to use only a subset for reconstruction and rendering, and to generate other virtual views at positions that are also available in the test set. This enables objective and subjective comparison of texture and shape rendering quality. However, suitable quality measures still need to be defined. It is questionable that PSNR is a good measure for virtual intermediate views, since small misalignments might cause large errors that are visually often not relevant. In addition, measures for shape reconstruction and other distortions within the objects (e.g. along epipolar lines) need to be developed [15]. So far, only the visual impression is appropriate.

The experiments were performed with several data sets made available to the MPEG 3DAV group. As illustrated in Fig. 1, humans in motion were captured in a dome-like multi-camera setting. The geometry was reconstructed from the provided silhouette images as explained in section 2. The resulting wire frames were encoded using MPEG-4 mesh coding [4], which allows varying the number of bits used per vertex. The resulting bitrates using the Doo Young data set are shown for some settings in

(2000 vertices/mesh). Examples of decoded meshes are shown in Fig. 4. We have found that increasing the number of bits per vertex above 8 bits does not significantly increase visual quality, when the texture is mapped onto the mesh.

Current MPEG-4 mesh coding only allows to encode the meshes for each frame separately (corresponding to I-frame coding in video). However, the meshes describe the same object moving over time, and therefore the compressed data still contain a huge amount of temporal statistical dependency. Our future research will therefore be to develop predictive coding for moving and deforming 3D meshes. We also plan to constrain the geometry reconstruction in order to produce more time-consistent meshes, since this process also still works independently for each time instant.

Tab. 1: Bit rates for geometry in kbit/s

bit/vertex	6	7	8	10
bitrate [kbit/s]	556,42	685,33	831,08	1088,25

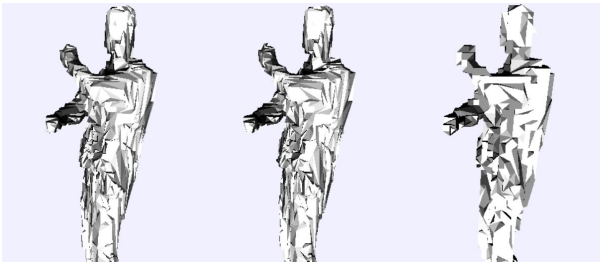

Fig. 4: Original mesh (left) and decoded meshes at 8 (middle) and 6 (right) bit per vertex

In general, the quality of view-dependent texture mapping increases with the number of textures used. On the other hand, also the data rate and the processing and rendering complexity increase. We therefore tested our algorithm with subsets including 4 and 8 video textures that were encoded using H.264/AVC at 3 different bit rates (256, 128, 64 kbit/s/view).

Fig. 5 compares details of virtual rendered views in the middle between original views. The left example was rendered with uncompressed textures. The other images show examples with textures coded at 128 and 64 kbit/s/view illustrating the degradation of image quality with reduced bit rate.

Fig. 5: Details of rendered views uncoded, 128 and 64 kbit/s/view

Fig. 6 compares rendered views generated using 4 and 8 textures at the same total texture bit rate of 512 kbit/s. This means that the 4-view set was coded at 128 kbit/s/view and the 8-view set was coded at 64 kbit/s/view. Rendering artifacts decrease with the number of views, however, coding artifacts increase with the number of views. This illustrates that for a

free viewpoint video application, a tradeoff has to be found regarding the number of views for optimum overall visual quality at a particular given bit rate.

Fig. 6: Rendered views using 4 and 8 textures at same total texture bitrate of 512 kbit/s

The functionality of free viewpoint video is illustrated in Fig. 7. It shows 4 rendered views at 4 different time instances from 4 different virtual viewpoints. These are snapshots from a virtual camera fly around the object as it moves. The complete video sequences as well as other examples can be downloaded from [1].

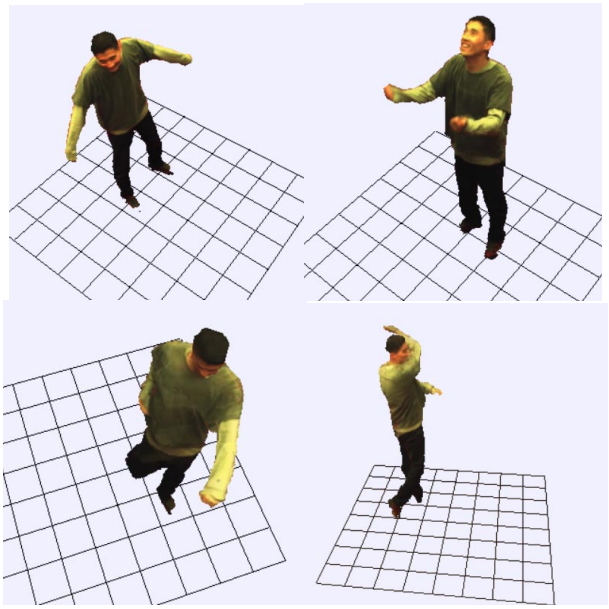

Fig. 7: Virtual camera fly, rendered views at 4 different times from 4 different virtual viewpoints

6. SUMMARY AND FUTURE WORK

The presented system for free viewpoint video covers the entire processing and transmission chain from cameras to display including data compression. Our results indicate suitability of the tested approaches for the envisaged application. Predictive mesh coding to exploit temporal redundancy and constrained geometry reconstruction for time consistent mesh generation are subject to further research. Also improved H.264/AVC coding taking into account the object shape will be studied in the future.

ACKNOWLEGEMENT

We would like to thank the Computer Graphics Lab of ETH Zurich for providing the Doo Young multi-camera data set.

REFERENCES

[1] S.B. Kang, R. Szeliski, and P. Anandan, "*The Geometry-Image Representation Tradeoff for Rendering*", Proc. ICIP, Vancouver, Canada, September 2000.

[2] P. Eisert, E. Steinbach, and B. Girod, "*Multi-hypothesis, Volumetric Reconstruction of 3-D Objects from Multiple Calibrated Camera Views*", Proc. ICASSP, pp. 3509-3512, Phoenix, Mar. 1999.

[3] W. E. Lorensen, and H. E. Cline, "*Marching Cubes: A high resolution 3D surface reconstruction algorithm,*" Proc. SIGGRAPH, vol. 21, no. 4, pp 163-169, 1987.

[4] ISO/IEC JTC1/SC29/WG11, "*Information Technology - Coding of Audio-Visual Objects, Part 2: Visual; 2001 Edition,*" Doc. N4350, Sydney, Australia, July 2001.

[5] ITU-T Recommendation H.264 & ISO/IEC 14496-10 AVC, "*Advanced Video Coding for Generic Audio-Visual Services,*" 2003.

[6] C. Buehler, M. Bosse, L. McMillan, S. Gortler, and M. Cohen, "*Unstructured Lumigraph Rendering,*" Proc. SIGGRAPH, pp. 425-432, 2001.

[7] K. Mueller, and A. Smolic, "*Study on View-Dependent Multitexturing for MPEG-4 AFX,*" ISO/IEC JTC1/SC29/WG11, MPEG03/M10152, Gold Coast, Australia, October 2003.

[8] R.Y. Tsai, "*A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV camera and lenses,*" IEEE Journal of Robotics and Automation, Vol. RA-3, No. 4, August 1987.

[9] K. Mueller, A. Smolic, M. Droese, P. Voigt, and T. Wiegand, "*Multi-Texture Modelling of 3D Traffic Scenes,*" Proc. ICME, Baltimore, MD, USA, July 6.-9. 2003.

[10] R. Szeliski, "*Rapid Octree Construction from Image Sequences,*" CVGIP: Image Understanding, Vol. 58, No. 1, July, pp. 23-32, 1993.

[11] P. Debevec, C. Taylor, and J. Malik, "*Modeling and rendering architecture from photographs: A hybrid geometry- and image based approach,*" Proc. SIGGRAPH, pp. 11-20, 1996.

[12] D. Vlasic, H. Pfister, s. Molinov, R. Grzeszczuk and W. Matusik, "*Opacity Light Fields: Interactive Rendering of Surface Light Fields with View-dependent Opacity*", Proc. 2003 Symposium on Interactive 3D graphics, pp. 65-74, 2003.

[13] A. Smolic, and D. McCutchen, "*3DAV Exploration of Video-Based Rendering Technology in MPEG,*" IEEE Trans. on CSVT, Vol. 14, No. 9, pp. 348-356, March 2004.

[14] ISO/IEC JTC1/SC29/WG11, "*Text of ISO/IEC 14496-16:2003/FDAM4,*" Doc. N5397, Awaji, Japan, December 2002.

[15] M. Rittermann, "*Quality Assessment of 3D Video Objects,*" IEEE/ISCE'03, Sydney/Australia, 3rd-5th December 2003.

[16] http://bs.hhi.de/~smolic/ICIP04.html