Unconstrained Free-Viewpoint Video Coding

Edouard Lamboray Stephan Würmlin Michael Waschbüsch Markus Gross Hanspeter Pfister*
Computer Graphics Laboratory, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland
**MERL - Mitsubishi Electric Research Laboratories, Cambridge, MA, USA*
{lamboray, wuermlin, waschbuesch, grossm}@inf.ethz.ch pfister@merl.com

Abstract

In this paper, we present a coding framework addressing image-space compression for free-viewpoint video. Our framework is based on time-varying 3D point samples which represent real-world objects. The 3D point samples are obtained after a geometrical reconstruction from multiple pre-recorded video sequences and thus allow for arbitrary viewpoints during playback. The encoding of the data is performed as an off-line process and is not time-critical. The decoding however, must support for real-time rendering of the dynamic 3D data. We introduce a compression framework which encodes multiple point attributes like depth and color into progressive streams. The reference data structure is aligned on the original camera input images and thus enables for easy view-dependent decoding. A novel differential coding approach permits random access in constant time throughout the entire data set and thus enables arbitrary viewpoint trajectories in both time and space.

1. Introduction

In recent years, free-viewpoint video has appeared as a new technology for interactive rendering of dynamic real-world objects from arbitrary viewpoints. The respective 3D information is usually obtained from multiple, synchronized 2D video streams. The MPEG-4 committee is currently investigating various methods for coding 3D audio/video data [3]. Many 3D reconstruction methods have been proposed but a complete free-viewpoint video pipeline including a compression stage of the time-varying 3D data still needs to be developed. Figure 1 shows a sketch of a free-viewpoint video acquisition stage with 16 cameras and an example of an arbitrary spatial path of the virtual viewpoint during playback.

In this paper, we present a compression framework for 3D video fragments which is a dynamic point based representation tailored for real-time streaming and display of free-viewpoint videos [12]. A 3D video fragment holds, additionally to its color, a number of geometrical attributes. Moreover, a one-to-one relation between 3D point samples and foreground pixels in the input 2D video images is guaranteed. 3D video fragments are generic in the sense that they work with any 3D reconstruction method which extracts depth from images. Thus the representation is quite complementary to model-based scene reconstruction methods using volumetric (e.g. space carving, voxel coloring), polygonal (e.g. polygonal visual hulls) or image-based (e.g. image-based visual hulls) scene reconstruction. Our framework can thus be seen as an abstraction of a 3D video representation and its compression from the 3D reconstruction methods.

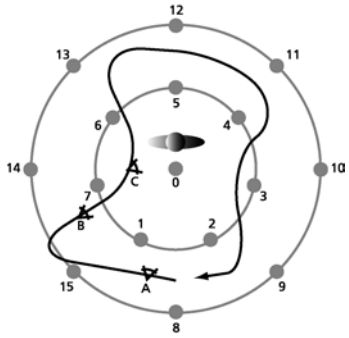


Figure 1: Top view of a free-viewpoint video acquisition stage. The black line illustrates an arbitrary spatial path of the virtual viewpoint during playback.

1.1. Related work

Several coding techniques for large but static point representations have been proposed in the literature [1, 7]. Botsch et al. report memory requirements between 8 and 13 bit per point for storing static point sampled geometry in an octree data structure including surface normal and color attributes. Time-varying 3D video data has been encoded by Wuermlin et al. at comparable bit rates [13]. Briceno et al. propose to reorganize the data from dynamic 3D objects into 2D images before deploying video compression techniques for coding animated meshes [2]. Vedula et al. developed a free-viewpoint video system based on the computation of a 3D scene flow and spatio-temporal view interpolation but did not address the coding of this representation [11].

2. Playback features

Depending on the desired features of a free-viewpoint video system and the need for real-time playback, a well designed compression scheme for time-varying 3D data should address the following features:

Multi-resolution. Scalability and progressivity with respect to resolution. This feature can be achieved using either progressive encoding or progressive sampling of the data [8, 12].

Multi-rate. Scalability with respect to time, i.e. the playback of the sequence is possible at a different frame rate than the recording frame rate. Backward playback should also be possible.

View-dependent decoding. Taking into account that the number of cameras is potentially large and that real-time decoding is required, it is not realistic on current hardware to

decode all the cameras before rendering the scene for a given viewpoint. Hence, it is necessary to reduce the set of processed cameras already during decoding. Thus, view-dependent decoding is an important characteristic of a real-time free-viewpoint video decoder.

We define view-dependent decoding such that for a given rendering frame, the decoder maximizes the ratio of decoded information which is finally rendered versus the total amount of decoded information for the given rendering instant. If the 3D video data is encoded in image-space, the technique described by Wuermlin et al. can be used for deciding which cameras are required for rendering from the current virtual viewpoint [12]. Thus, given a viewpoint and the camera calibration data, the decoder computes the contributing cameras and reads the data accordingly.

If we assume that data from the three cameras closest to the current virtual viewpoint is used to render the free-viewpoint video during playback, the example of Figure 1 leads to the following configuration: Viewpoint A requires cameras 1, 8, and 15; viewpoint B cameras 1, 7, and 15; viewpoint C cameras 0, 6 and 7.

This example illustrates that during the spatial navigation of the viewer, the set of contributing cameras is permanently changing. Hence, if view-dependent decoding is implemented and if the data is encoded in image-space, the stream from each camera must be randomly accessible and virtually every single frame needs to be an entry point to the stream. This requirement is definitely very restricting and we thus propose to distinguish between two classes of free-viewpoint video which reflect different degrees of spatio-temporal navigability.

Constrained free-viewpoint video. After decoding, the 3D data can be rendered from any possible direction, but either only small viewpoint changes are allowed during rendering, or discontinuities in rendering are tolerated in presence of large viewpoint changes.

Unconstrained free-viewpoint video. After decoding, the 3D data can be rendered from any possible direction, the viewpoint being a function of the rendering time and the discontinuities during rendering are minimized.

According to the above definitions, the spatio-temporal viewpoint trajectory during playback of unconstrained free-viewpoint video can be completely arbitrary. In the constrained case, the trajectory is restrained to a narrow band, as illustrated in Figure 2.

3. Data acquisition and preprocessing

The input data of free-viewpoint video systems acquiring real-world objects typically consists of multiple concentric video sequences of the same scene, which are recorded with synchronized cameras. The cameras are calibrated and intrinsic and extrinsic calibration parameters are available to both encoder and decoder. Furthermore, a segmentation mask needs to be provided for every input frame. The segmentation mask tells which pixels belong to the object of interest, i.e. are foreground pixels.

After the input images have been processed by an appropriate 3D reconstruction algorithm, each foreground pixel has associated depth and color values. In combination with the camera calibration parameters, the depth values describe the geometry of the object. In general, it is possible to additionally encode any attributes describing the visual appearance of an object. The set of optional attributes essentially depends on the final rendering scheme.

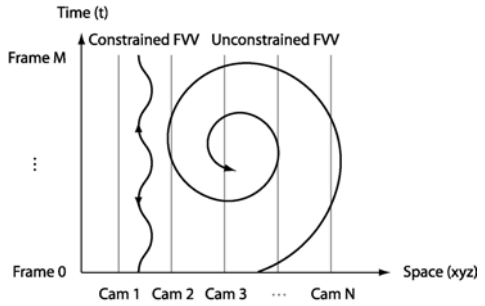


Figure 2: Possible viewpoint trajectories for constrained and unconstrained free-viewpoint video.

In summary, the compression framework described in this paper can be applied to every static or dynamic 3D data set which can be completely described by a set of concentric 2D views. In practice, this applies to many – if not all – acquisition setups for 3D reconstruction of real world objects. Moreover, our coding framework can smoothly be extended to an arbitrary camera setup where only a few cameras see the object at a time.

4. Image-space free-viewpoint video coding

The underlying data representation of our free-viewpoint video format is a dynamic point cloud. Since the point attributes are separately stored and compressed, a referencing scheme, allowing for the unique identification between points and their attributes is mandatory. Using the camera images as building elements of the data structure, each point is uniquely identified by its position in image space and its camera identifier. Furthermore, looking separately at each camera image, we are only interested in foreground pixels, which contribute to the point cloud describing the 3D object.

Thus, we use the segmentation mask from the camera images as reference for all subsequent coding schemes. In order to avoid shifts and wrong associations of attributes and points, a lossless encoding of the segmentation mask is required. This lossless segmentation mask must be at the disposal of all encoders and decoders. However, all pixel attributes can be encoded by a lossy scheme. Nevertheless, a lossless or almost lossless decoding should be possible if all data is available.

The overall compression framework is depicted in Figure 3. From the segmentation masks and the camera calibration data, a geometric reconstruction of the object of interest is computed. The output of the geometric reconstruction are 3D positions. The data streams are compressed and, along with the texture information and the segmentation masks, multiplexed into an embedded, progressive free-viewpoint video stream. The camera calibration data is encoded as side information.

In the remaining of this paper, we discuss more specifically image-space encoding schemes for the point attributes. For shape coding, we rely on lossless coding techniques for binary images [4, 6] where the compression is based on context-sensitive adaptive binary arithmetic coders.

5. Unconstrained free-viewpoint video

Conventional video codecs exploit motion prediction in subsequent frames and encode them in a recursive way. They produce a stream consisting of reference frames – which are used as entry points into the stream – predicted frames and interpolated frames. Obviously the predicted and interpolated

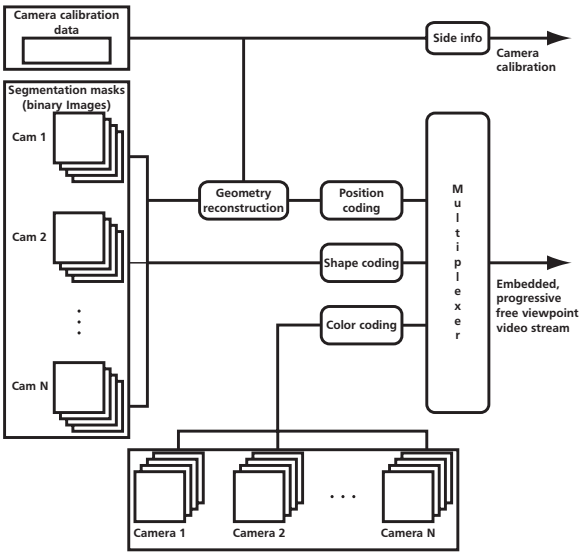


Figure 3: Compression framework at the encoder.

frames achieve a higher compression ratio than the independent reference frames. However, for unconstrained free-viewpoint video the encoded stream must allow for arbitrary entry points and thus the number of frames between two reference frames must be relatively small.

State-of the art research in video coding also addresses the problem of scalable video coding where scalability is understood in terms of image resolution and frame rate [10, 14]. However, the target application of current research and standardization efforts in scalable video coding is scalable playback of conventional 2D video. For example, the criteria on temporal scalability suggests possible playback rates for at least three defined frame rates [9]. Such an approach does not automatically lead to true random access in constant time with arbitrary entry points and hence does not completely fulfill the requirements of unconstrained free-viewpoint video coding.

Average coding. In this section, we propose a coding scheme that implements arbitrary entry points into a 2D video stream and builds upon a reference and a delta frame using the prediction from the reference frame. We suggest to use temporally averaged information in the reference frame and encode the difference between the original frame and the reference frame in the corresponding delta frame. Thus, a reference frame is valid for a specified time window, i.e. N frames, and is generated by computing the average value of each attribute for every foreground pixel. Note that – within the respective time window – the reference frame is independent from the recording time. Furthermore, for every window of N frames which use the same reference frame, we need to encode $N+1$ frames in total. However, we expect the additional cost of coding the reference frame to be distributed over a large number of actually displayed frames. Furthermore, a rough approximation of the average frame is sufficient, since the image details are averaged out anyway. The principle of average coding is illustrated in Figure 4. Figure 5 depicts average coding using texture data from one single camera.

Note that in free-viewpoint video coding we are essentially interested in coding single objects and not full video frames. The suggested approach would most probably fail for coding full-frame video streams.

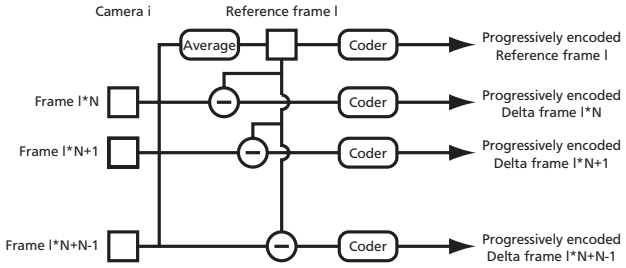


Figure 4: Principle of average coding

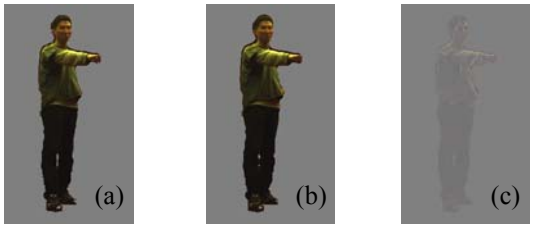


Figure 5: Average coding illustrated with a texture example. a) Masked camera input image; b) Masked reference frame; c) Masked delta frame.

6. Results

For the actual coding of colors and depths we currently use zero-tree wavelet transform coding [8], followed by arithmetic coding. In Figure 6, we compare the PSNR of the average encoded texture frames to single frame JPEG2000 encoding. The results show that average coding is superior to single frame coding for most time windows throughout the sequence. The increase of the reference time window from 5 to 15 frames affects the PSNR performance by approximately -0.5 dB. For the parts of the sequence with a high motion activity, i.e. in between frames 150 and 170, the single frame encoding is more efficient. We thus suggest to use a hybrid coding scheme for unconstrained free-viewpoint video: Average coding, potentially with an adaptive window size, should be used during periods of low motion activity; otherwise, single frame coding should be used.

In this experiment, we used a real-world input data set with a resolution of 640x480 pixels at 25 frames per second. Figure 5 shows example frames from this data set.

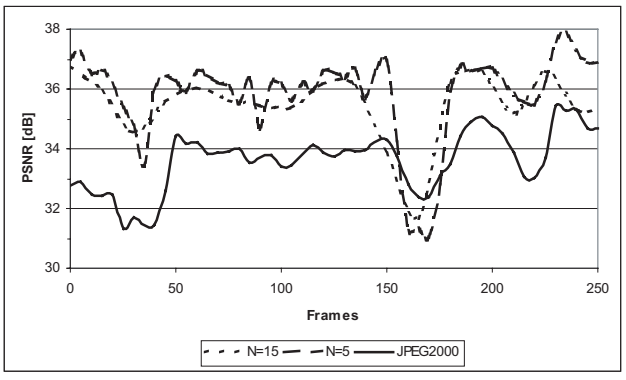


Figure 6: Zero-tree average coding for two different window sizes compared to JPEG2000 plain image compression.

Alternatively to the coders compared in Figure 6, MPEG-4 video object coding, which is based on shape adaptive discrete cosine transform coding, could be used. However, MPEG-4 does not yet allow for progressive decoding [5].

Although free-viewpoint video coding aims at coding of real-world objects, we also use a test sequence generated from a synthetic model. The data however is processed identically to real-world data, i.e. the synthetic model is observed from multiple virtual cameras which provide the input data to the free-viewpoint video pipeline. The input images of this sequence have a resolution of 320x240 at 25 frames per second. The use of a synthetic test sequence further allows to ignore practical problems of data acquisition and camera calibration while working on the compression issues. Additionally, only a synthetic model can provide exact reference values for the geometry attributes.

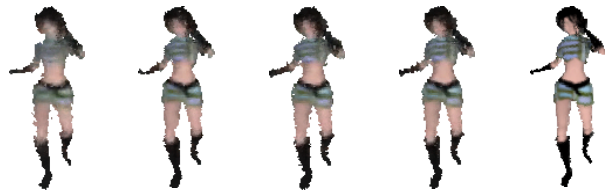
The fair evaluation of free-viewpoint video formats is however a difficult task and no appropriate error metric has yet been defined. The traditional metrics like PSNR do not deliver reasonable results if applied to result images rendered from viewpoints lying in between two or more input camera positions.

In the experiment of Table 1, we compressed the test data sequence with our unconstrained free-viewpoint video codec using one single average frame for the complete sequence, i.e. the reference frame covers 200 frames which corresponds to 8 seconds at normal playback speed. We further improved the compression performance by downscaling the depth images and re-expanding them to full resolution during decoding.

For view-dependent decoding and rendering, we select the three cameras closest to the virtual viewpoint. The resulting 3D object consists of approximately 8000 point samples per frame. We progressively decode our data with several example bit rates as specified in Table 1. The total bit rate includes the contribution of reference and delta frames for depth and texture and the lossless shape coding. Snapshots from the respective sequences are shown in Figure 7.

Table 1: Example bit rates for unconstrained free-viewpoint video with three reconstruction cameras per virtual viewpoint.

Depth image resolution	Depth bits / sample	Color bits / sample	Total bit rate bits / second
107x80	0.166	0.7	350k
160x120	0.5	1.5	560k
320x240	1.5	1.5	740k
320x240	2.0	2.0	920k
320x240	8.0	24.0	10200k (uncompressed)



350 kbps 560 kbps 740 kbps 920 kbps uncompr.

Figure 7: Example renderings at various bit rates for a virtual viewpoint in between three cameras images.

7. Conclusions and outlook

This paper introduces a compression framework for free-viewpoint video using a point-based data representation. The deployment of a novel average coding scheme enables for an unconstrained spatio-temporal navigation throughout the 3D video stream. Ideally, all data is progressively encoded and hence a free-viewpoint video stream can be generated for different target bit rates.

In the future, the specific codecs to be used in our compression framework need to be investigated in detail. Furthermore, an algorithm optimizing the window length for average coding should be developed. Finally, we would like to investigate how the proposed free-viewpoint video coding scheme can be implemented in standardized multimedia frameworks like MPEG-4.

Acknowledgements

The authors would like to thank the GrOVis group of the Max-Planck-Institut für Informatik in Saarbrücken for providing the synthetic test sequence and Aljoscha Smolic for many fruitful discussions.

References

[1] M. Botsch, A. Wiratanaya, and L. Kobbelt. Efficient high quality rendering of point sampled geometry. In *Proceedings of the 13th Eurographics Workshop on Rendering*, pages 53–64, 2002.

[2] H. Briceno, P. Sander, L. McMillan, S. Gortler, and H. Hoppe. Geometry videos: A new representation for 3d animations. In *Proceedings of ACM Symposium on Computer Animation 2003*, pages 136–146, July 2003.

[3] Description of explaration experiments in 3DAV. ISO/IEC JTC1/SC29/WG11 N6194, December 2003.

[4] A. K. Katsaggelos, L. P. Kondi, F. W. Meier, J. Ostermann, and G. M. Schuster. MPEG-4 and rate-distortion-based shape-coding techniques. *Proceedings of the IEEE*, 86(6):1126–1154, June 1998.

[5] J. Ostermann, E. S. Jang, J.-S. Shin, and T. Chen. Coding of arbitrarily shaped video objects in MPEG-4. In *Proceedings of ICIP*, pages 496–499, 1997.

[6] W. B. Pennebaker, J. L. Mitchell, G. G. Langdon, and R. Arps. An overview of the basic principles of the q-coder adaptive binary arithmetic coder. *IBM Journal of Research and Development*, 32(6):717–726, 1988.

[7] S. Rusinkiewicz and M. Levoy. QSplat: A multiresolution point rendering system for large meshes. In *SIGGRAPH 2000 Conference Proceedings*, ACM Siggraph Annual Conference Series, pages 343–352, 2000.

[8] J. M. Shapiro. Embedded image coding using zerotrees of wavelet coefficients. *IEEE Transactions on Signal Processing*, 41:3445–3462, December 1993.

[9] Requirements and applications for scalable video coding. ISO/IEC JTC1/SC29/WG11 N6052, October 2003.

[10] D. Taubman and A. Seeker. Highly scalable video compression with scalable motion coding. In *Proceedings of ICIP*, volume 3, pages 273–276, 2003.

[11] S. Vedula, S. Baker, and T. Kanade. Spatio-temporal view interpolation. In *Proceedings of the 13th Eurographics Workshop on Rendering*, June 2002.

[12] S. Wuermlin, E. Lamboray, and M. Gross. 3D video fragments: Dynamic point samples for real-time free-viewpoint video. *Computers & Graphics, Special Issue on Coding, Compression and Streaming Techniques for 3D and Multimedia Data*, 28(1), 2004.

[13] S. Wuermlin, E. Lamboray, O. G. Staadt, and M. H. Gross. 3D video recorder. In S. Coquillart, H.-Y. Shum, and S.-M. Hu, editors, *IEEE Pacific Graphics 2002 Proceedings*, pages 325–334. IEEE Computer Society Press, October 2002.

[14] Z. Zhang, G. Liu, and Y. Yang. High performance full scalable video compression with embedded multiresolution MC-3DSPIHT. In *Proceedings of ICIP*, volume 3, pages 721–724, 2002.