NEAR-FUTURE STREAMING FRAMEWORK FOR 3D-TV APPLICATIONS

Goran Petrovic

Eindhoven University of Technology
5600 MB Eindhoven, The Netherlands
g.petrovic@tue.nl

Peter H. N. de With

LogicaCMG / Eindhoven Univ. Technol.,
PO Box 7089, 5605 JB Eindhoven, Netherlands
P.H.N.de.With@tue.nl

ABSTRACT

This paper presents a layered framework for 3D-TV applications, combining multiview and depth-image based approaches in a scalable fashion. To solve the problem of missing data due to disocclusions, we add specific layers for coded occlusion data and the edge-mask information for high-quality 3D rendering of key objects in the scene. We show how the same framework can be extended towards FTV applications by jointly addressing simulcast and multicast transmission. By adopting a distributed delivery architecture, new interesting properties can be realized such as shared processing for the creation and streaming of virtual viewpoints.

1. INTRODUCTION

In an attempt to anticipate future deployment of 3D-video systems [1], the MPEG community has recently singled out two application scenarios: *Three-Dimensional Television (3D-TV)* and *Free Viewpoint Television (FTV)*, and one enabling technology - *Multi-View Video Coding (MVC)* [1]. We find these recommendations adequate in that they point the ways in which more realism can be added to the conventional TV and video systems. A recent survey of MVC standardization activities [1] illustrates a strong bias towards existing standards (e.g., H.264). Although this viewpoint can be understood, we believe that the opportunity exists to further explore combinations of video data, scene geometry information, scalability and interactivity to improve the overall framework and pursue a broadly applicable architecture. These combinations will be discussed in more detail in this paper.

The 3D-TV application enables a viewer to perceive depth in the displayed scene. Two closely spaced images of the same scene are displayed simultaneously to create the effect of depth. This is a well-known concept of *stereoscopic* video, which 3D-TV extends from the service perspective, by defining a suitable infrastructure for broadcasting such content to the users. With FTV, a scene can be displayed from different viewpoints in an *interactive* fashion. A user either selects an arbitrary new viewpoint and a viewing direction, or the user's movements are continuously tracked and the displayed content automatically adjusted to the new position.

Stereoscopic and multi-perspective scene viewing require new approaches for content acquisition, coding, transmission and display [1]. Our aim in this paper is to explore the design space for 3D-TV and FTV services, while focusing on the content transmission and delivery aspects. The main motivation in doing so is that the transmission system issues currently receive little attention in 3D-video research, although equally important as the corresponding 3D content modeling and recording techniques. The position we take is that the IP-based networks are best positioned to serve as a substrate for the gradual deployment of 3D-TV and FTV services, and also as their long-term operational environment. We justify our position on IP-based networks with the following arguments: (1) Internet is where the interactive applications have their natural place; (2) Service deployment over the Internet opens access to a large client base, including a growing number of mobile users, which is important for the service acceptance; (3) Internet end-nodes are equipped with programmable processors, and algorithms implementing new functionalities can be realized in software, thereby facilitating service deployment. Based on these observations, we first address the following question: *which of the 3D representation formats reported in the literature is best suited for near-term deployment in the current Internet?* To this end, we propose a solution which is based on Depth Image-Based Rendering (DIBR) [2], and extend it with explicit disocclusion-filling information. We then focus on a problem which can be formulated as follows: *which service model best accounts for a highly heterogeneous client-base in today's IP networks in a scalable fashion?* As our second contribution, we propose a solution based on resource sharing in groups of collaborating network hosts.

The rest of the paper is structured as follows. In Section 2 we survey different formats for 3D video representation. Section 3 discusses the system design from a practical standpoint and motivates our choice of data format and communication system architecture. In Section 4 we suggest the ways in which our framework can be extended to allow the system to scale to a large number of concurrent, heterogeneous users. In Section 5 we present preliminary results. Section 6 concludes the main points of the presented framework.

2. BACKGROUND ON 3D-TV AND FTV

Shum *et al.* [3] give an overview of the approaches for representing real-world 3D scenes and rank those approaches depending on the amount of geometric information about the scene they recover. We adopt their general concept, but confine ourselves to considering dynamic scenes only.

One approach for 3D scene modeling is to construct the accurate 3D-shape or some other *volumetric model* for a number of key objects in the scene. The idea behind the approach is to extrapolate such a model from the available camera images and then use it to render arbitrary views of the same scene. Most commonly, a scene with a single human figure is considered. Aiming at real-time rendering, Matusik *et al.* [4] recover an incomplete, view-dependent visual hull model of a human object off-line, distributing the processing over several PCs. Rendering results show a good quality, while the hulls are extracted with as few as four cameras. However, rendering is complex and runs in parallel over four PCs to obtain real-time properties. Würmlin *et al.* [5] also take an approach based on visual-hull reconstruction, but use dynamic-point samples as the rendering primitive, instead of triangular meshes. Experiments show convincing results for rendering of human actors, captured with eight cameras. Carranza *et al.* [6] assume that a generic human-body model in the form of a triangular mesh is available, and focus on capturing the object's motion in the scene by tracking this model throughout the sequence. For rendering, view- and time-dependent textures captured by the cameras are mapped onto the model for rendering at video rates; however, the motion capture is performed off-line.

An alternative approach is to render multiple views of a dynamic scene from the input images directly [7], or compute a basic scene-geometric information (e.g., a *depth map*) to reduce the number of input images or improve the rendering quality. Fehn *et al.* [2] propose to use depth streams for the narrow-field view interpolation in 3D-TV broadcast. Geometry information in a depth stream involves a depth value for each pixel in every frame of the original stream. Multiple nearby views of the scene are generated at the receiver by re-projecting the original pixels into the new viewpoint, based on the depth map.

It can be deduced from the above that different views on 3D processing exist, but the evaluation of their relative merits in the absence of directly comparable data is difficult. In such a case, it is usually beneficial to define a framework that attempts to combine the attractive points of individual approaches. It can be readily motivated that although volumetric object models may be attractive in advanced professional applications, this would bring little or no benefit for the 3D-TV applications in the broadcast/simulcast case, where the user is after a 3D experience at an affordable cost (consider e.g., rendering complexity at the receiver). Therefore, we believe that the goal can be better achieved by combining the multi-view and depth-based approaches and pursuing *scalability* as an architectural property.

In summary, techniques that extract global object geometry, like the advanced volumetric approaches, have the potential to significantly reduce the number of images, which is attractive for IP video streaming applications. However, current techniques for global geometry extraction and rendering are not likely to be a cost-effective solution for real-time system operation in the near future. Furthermore, such techniques scale poorly with the scene complexity (e.g., the number of objects in the scene). For this reason, we exclude them from the near-term solution.

3. TRANSMISSION FRAMEWORK FOR 3D-TV

Depth map is efficient. When considering that image-based representations have high data volumes, broadcasting all available camera streams is impractical. A depth-map is an efficient alternative for narrow-angle viewpoint changes, as it can be used to re-project the corresponding intensity data to a newly chosen camera plane, in a process known as *warping*.

Layered extension to multiview streaming. For sufficient scalability, we pursue a layered framework where the number of views and the associated depth-maps received can be extended dynamically and on-demand.

Addition of occluded data. If a wide-angle perspective change is requested, rendering artifacts come in the form of holes in the texture of the synthesized views. This situation illustrates the "dissoclusion" problem. The solutions to this problem are occlusion-compatible warping and filling the holes by background extrapolation [2]. For large occlusions, this approach will have its limitations. In our IP-networked case, we propose to add a specific stream that provides the occluded data explicitly.

Interactivity. Broadcasting all the layers for every available viewpoint is inefficient (although attractive for FTV applications as it allows the clients to quickly switch between the available viewpoints) in that it treats all the streams as equally important, while in real sessions, some streams will be requested frequently, and others not at all. A better design is to schedule new viewpoint transmissions *reactively* and *on-demand*.

The framework of Fig. 1 shows a more detailed view on our proposal. Apart from streaming multiple texture and depth streams, it can be seen that two layers have been added: edge and occlusion information. The former provides more accurate information on key objects for high quality display [8]. All layers can be filled and coded in a scalable fashion. Additionally, a practical multi-stream framework must implement inter-stream synchronization and joint packet scheduling, congestion control, and optionally error control functions with unreliable transport protocols (e.g., UDP). The rest of the framework is based on existing streaming technology (RTP/RTCP, RTSP, SDP) and is not further discussed here.

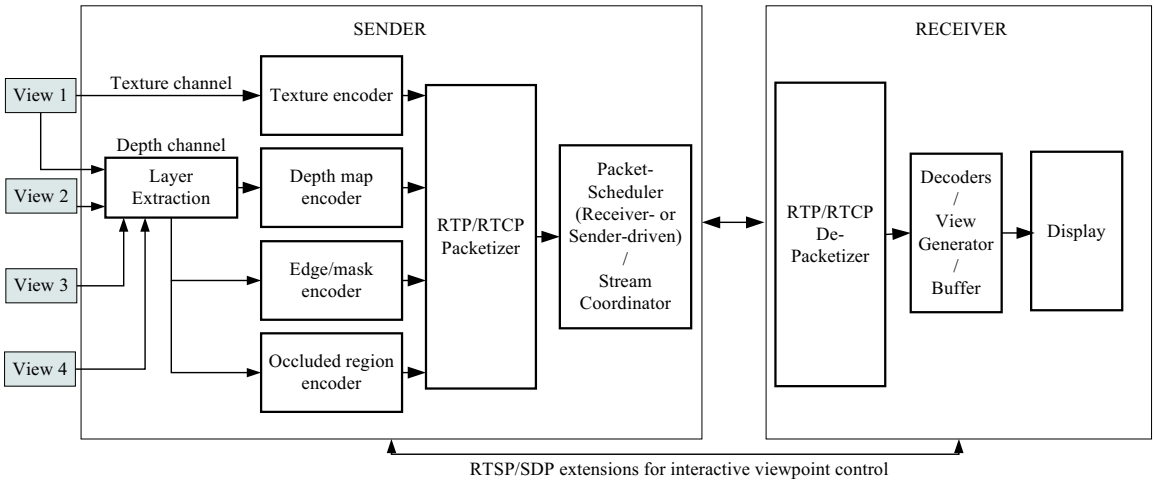

Fig. 1. 3D streaming framework.

4. LARGE-SCALE 3D-CONTENT DELIVERY

In this section we address the performance issues of 3D-TV (FTV) systems implemented within the framework of Section 3, for the case of a large number of concurrent, heterogeneous users. Large data volumes in multi-camera systems coupled with the processing for interactive viewpoint adaptation pose new challenges to the design of scalable multimedia servers and delivery architectures.

To better understand the design issues involved, we consider a scenario where 3D-TV and FTV technologies are deployed to enhance live broadcast. We make the following two assumptions: (1) the event is captured by multiple, fully calibrated cameras; (2) each camera is assigned a unique logical identifier providing an interface to its calibration parameters, the associated depth map, edge mask and occlusion information. Our layered model provides basic support for heterogeneity of display devices and client access bandwidths. All viewers receive a 2D stream for their camera angle of choice, while those equipped with 3D display devices also receive the additional layers associated with that camera angle (*3D-streams*). Optionally, 2D-viewers can receive full 3D streams to perform narrow-angle viewpoint changes. In general, wide-angle viewpoint changes are supported by switching to the desired camera angle. A special case of a wide-angle viewpoint change is the viewpoint case that does not match any of the physical cameras, nor can it be created using the available depth and occlusion-handling layers. In this case, a *virtual viewpoint* needs to be constructed combining a number of original camera streams (and the associated additional layers). Two options for implementing this functionality exist. Either a server computes the desired viewpoint and streams the result to the client, or the server sends the original streams and the client does the reconstruction. The choice between the two is a trade-off between the computation and network bandwidth.

4.1. Bandwidth cost

Conceptually, IP-multicast provides for an efficient usage of both the server-bandwidth and the wide-area network bandwidth, thus improving scalability of video streaming systems compared to the unicast case. A 3D-TV (FTV) server employing multicast delivery starts one multicast session for each requested 2D camera stream and one for each accompanying additional stream. Every client selects a number of multicast transmissions to receive, based on its preferences or capabilities. This way, the server-bandwidth cost depends on the number of active viewpoints, but is independent of the number of clients. Still, global support for IP-multicast remains limited, and the accessible client base is small [9]. This is a serious concern for 3D-TV and FTV applications aiming to grow to a TV broadcast-scale service and alternative delivery methods need to be investigated.

4.2. Server-side processing cost for interactivity

From the service perspective, processing for virtual viewpoint generation is best implemented at the server, thus reducing the bandwidth and processing costs at the client. However, serving a large number of such requests concurrently quickly consumes computational resources on the server, and a more scalable solution is required.

4.3. Developing the case for resource sharing

Motivated by the recent proposals for synchronous [9] and asynchronous [10] content delivery using network overlays, we illustrate the concept of resource sharing for reducing the bandwidth and processing costs in large scale 3D-TV and FTV systems (Fig. 2).

A 3D-TV (FTV) server transmits two different camera angles (3D-streams) to Nodes 1 and 4 respectively, using unicast connections to alleviate the lack of IP-multicast routing be-

tween different ISPs. Nodes 1 and 4 in turn relay the streams to other nodes on their IP multicast-enabled networks. Joining the overlay, Node 7 (3D-viewer) acts as a proxy for supplying 2D stream to a resource-constrained Node 8. Suppose Node 9 contacts the server with a request to join an ongoing live transmission. The server responds by sending the address of a nearby node which is already receiving the desired viewpoint (Node 7), which adds the Node 9 to its list of served clients. Next, suppose an event occurs which generates intensive interest from multiple viewers (e.g. a goal in a soccer match). Node 8 desires to receive an instant replay of the event from a virtual viewpoint. The server cannot fulfill this request due to a flash crowd effect. Instead, it responds by sending the addresses of two nodes that received and cached the 3D-streams required for interpolation (Nodes 7 and 5). Node 5 sends its cached data to Node 7, thus allowing Node 7 to interpolate a novel view and serve Node 8. Summarizing, the first example shows bandwidth sharing, whereas the second one illustrates distributed processing.

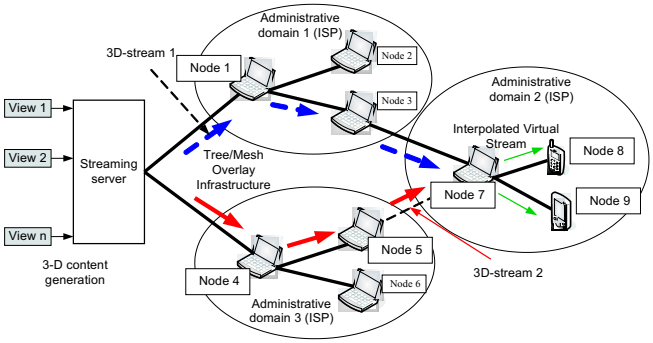

Fig. 2. Distributed streaming and virtual viewpoint creation.

5. PRELIMINARY EXPERIMENTS

We implemented the streaming test-bed from Section 3 between two nodes on a 100Mb/s LAN. Both the sender and the receiver were implemented on a desktop PC running Linux OS. We streamed the "Interview" test sequence and the associated depth-map sequence [2] at 25fps and 720×576 resolution. Both sequences were compressed in Simple Profile (SP) configuration using an MPEG-4 Reference Encoder and stored in separate files. At the start of the session, the compressed MPEG-4 elementary streams were packetized as specified in RTP (IETF RFC 3016). The RTP/RTCP protocol stack implementation was provided with "Live555 Streaming Media" library. The sender transmitted texture and depth frames in succession. Different UDP ports for texture and depth streams were used both at the sender and the receiver. The decoding at the receiver involved two independent MPEG-4 decoder processes, and the decoded frames were buffered prior to display. To test the end-to-end data pipeline, our player used the depth maps to compute and display an anaglyph sequence. The playback was smooth and the depth cues were well visible, even when viewed with inexpensive R/B filter glasses. This experiment is now extended by adding more streams for high-performance depth-map extensions containing occlusion data or adding secondary views to support multiview approaches.

6. CONCLUSION

This paper presents a layered framework for 3D-TV transmission, combining multiview and depth-based approaches in a scalable fashion. Besides texture and depth information, specific layers are added for coded occlusion data and edge-mask information to allow high-quality 3D rendering of key objects in the scene. By relying on a distributed delivery architecture and the concept of resource sharing for the creation and streaming of virtual viewpoints in a network overlay, we extend the range of viewpoints selectable by the user (FTV).

7. REFERENCES

[1] A. Smolic and P. Kauff, "Interactive 3D video representation and coding technologies," *Proceedings of the IEEE*, vol. 93, no. 1, pp. 98–110, Jan. 2005.

[2] C. Fehn, P. Kauff, M. Op de Beeck, F. Ernst, W. A. IJsselsteijn, M. Pollefeys, L. Van Gool, E. Ofek, and I. Sexton, "Evolutionary and optimised approach on 3D-TV," in *Proc. Int. Broadcast Conf. (IBC)*, Sept. 2002, pp. 357–365.

[3] H.Y. Shum, S.B. Kang, and S.C. Chan, "Survey of image-based representations and compression techniques," *IEEE Trans. Circuits & Systems Video Technol.*, vol. 13, no. 11, pp. 1020–1037, Nov. 2003.

[4] W. Matusik, C. Buehler, R. Raskar, S. Gortler, and L. McMillan, "Image-based visual hulls," in *Proc. Comp. Graphics (SIGGRAPH'00)*, July 2000, pp. 369–374.

[5] S. Würmlin, E. Lamboray, O.G. Staadt, and M.H. Gross, "3D video recorder," in *Proc. Pacific Conf. Comp. Graphics & Applic. (PG'02)*, Oct. 2002, pp. 325–334.

[6] J. Carranza, C. Theobalt, M. Magnor, and H.Seidel, "Free-viewpoint video of human actors," *ACM Trans. Graphics*, vol. 22, no. 3, pp. 569–577, July 2003.

[7] W. Matusik and H.-P. Pfister, "3D TV: A scalable system for real-time acquisition, transmission, and autostereoscopic display of dynamic scenes," in *Proc. Comp. Graphics (SIGGRAPH'04)*, Aug. 2004, pp. 814–824.

[8] L. Zitnick, S. B. Kang, M. Uyttendaele, S.Winder, and R. Szeliski, "High-quality video view interpolation using a layered representation," *ACM Trans. Graphics*, vol. 23, no. 3, pp. 598–606, Aug. 2004.

[9] A. Ganjam and H. Zhang, "Internet multicast video delivery," *Proceedings of the IEEE*, vol. 93, no. 1, pp. 159–170, Jan. 2005.

[10] S. Jin and A. Bestavros, "Cache-and-relay streaming media delivery for asynchronous clients," in *International Workshop on Networked Group Communication (NGC)*, 2002.