

Framework for Layered 3D Video Streaming

¹Goran Petrovic ^{1,2}Peter H.N. de With
¹Eindhoven University of Technology ²LogicaCMG Netherlands
P.O. Box 513 P.O. Box 7089
5600MB Eindhoven 5605JB Eindhoven
The Netherlands The Netherlands
g.petrovic@tue.nl 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. In an extensive system discussion, we conclude that it is possible to adaptively implement our streaming applications using a cross-layered approach. Similarly, the concept of selective reliability optimization can be introduced into our system.

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.

In our earlier paper [2], we argue 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. To this end, we propose a streaming solution which is based on Depth Image-Based Rendering (DIBR) [3], and extend it with explicit disocclusion-filling information. As our second contribution, we describe an efficient content delivery architecture 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 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 identifies challenges for future research and Section 7 concludes the main points of the presented framework.

2 Background on 3D-TV and FTV

Shum *et al.* [4] 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.

An important approach is to render multiple views of a dynamic scene from the input images directly [5], 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.* [3] 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 survey [4] 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 [4] 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 multiview 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 [2]. 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.

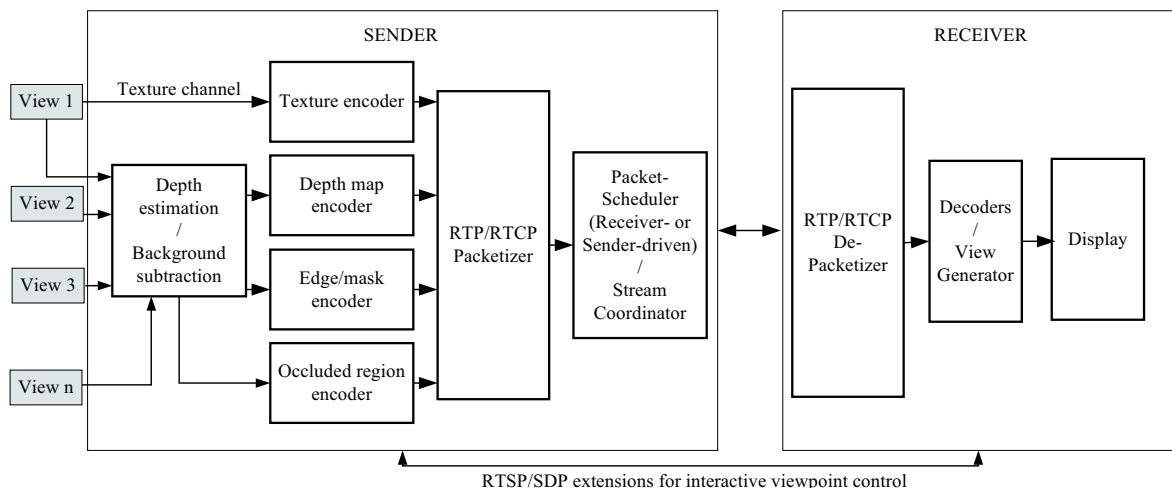


Figure 1: 3D streaming framework.

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 (Fig. 1).

Addition of occluded data and edge masks. 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 “disocclusion” problem. The solutions to this problem are occlusion-compatible warping and filling the holes by background extrapolation [3]. 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. Finally, the edge boundary for a number of key objects in the scene is useful, as it improves the quality of the interpolated views [6].

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 rest of the framework of Fig. 1 will be discussed in more detail in Section 6.

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 [7]. 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 [7] and asynchronous [8] 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).

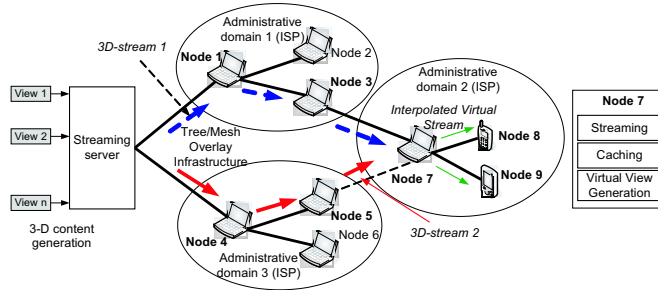


Figure 2: Distributed streaming and virtual viewpoint creation.

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 between 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.

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 [3] 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 [9]. 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 System aspects and considerations

In this section, we conceptually evaluate the streaming framework introduced in Sections 3 and 4 under practical constraints, and discuss the challenges it presents.

Extraction of layers. Since our focus in this paper is a streaming architecture, we currently rely on the availability of 3D test-sets provided by the MPEG 3DAV Working Group [1]. Each set includes a pre-extracted depth map and edge masks for key scene objects. However, in live 3D-TV and FTV sessions as in Section 4, these two layers need to be extracted at run-time. This is best implemented on original image data, thus prior to encoding (Fig. 1). For real-time operation, commercial live streaming systems commonly employ hardware encoders at the server back-end, and directly forward the compressed bitstreams to the clients via the server front-end [10]. Currently, few hardware devices are available that support real-time extraction of depth maps [3]. This leaves us with software processing for the time being, at an increased latency and server computational cost. For this reason, real-time operation and complexity are important factors in selecting an algorithm from the available options [11]. Similar considerations apply to the extraction of edge masks. Here, we will assume that a model of the scene background can be constructed off-line (e.g., in the case of a sports event), and focus on extracting key objects through background subtraction [12].

Streaming over best-effort networks. In contrast to the original (DIBR) proposal [3], intended for terrestrial and satellite TV broadcast, our framework primarily addresses best-effort IP networks. This raises concerns about the reliability and user-perceived QoS in 3D-TV (FTV) applications. From the transmission point of view, the Internet is a communication channel, shared statistically among multiple users. QoS parameters such as bandwidth, packet-loss rate and end-to-end delay are time-variant, reflecting the time-dependent traffic patterns in the network. With the inclusion of wireless last hops in the network, these parameters additionally depend on the varying conditions of the physical channel. Moreover, such time dependencies are generally unknown and virtually impossible to predict at the application design-time. A common solution for a streaming application to maximize its user-QoS under time-varying transmission conditions is to adopt *adaptivity* as a leading design principle [10]. Adaptive streaming attempts to keep the QoS levels in one dimension constant (e.g., frame rate), while compromising other QoS dimensions (e.g., quality, spatial resolution), based on the feedback from the network [13]. We believe that the 3D-TV applications implemented within our framework can substantially benefit from adaptivity. Particularly the image formation through warping (Section 3) introduces an interesting new QoS dimension for adaptivity. More specifically, it is important to study the effects of a QoS drop in one of the layers on the final presentation quality, and the possibility to compensate for those effects by allowing a higher QoS in other layers. Our framework also offers an opportunity for *reliability* optimizations that cut across its different layers of information. As Fig. 1 suggests, instead of transmitting each layer over a reliable end-to-end connection (e.g., with TCP), our current implementation relies on a connectionless, unreliable protocol (UDP). This decision is based on a common argument that the video streaming applications should trade full reliability for less rate fluctuation [10]. Previous research also demonstrates the advantages of providing the reliability in a selective way ([10], [14]). These approaches exploit the knowledge of the importance

of different bitstream parts for the presentation quality, and apply different levels of reliability based on this utility function (e.g., I- and P- frames in MPEG-4, base layer and enhancement layers for scalable video). We believe that an exciting opportunity exists to further develop these approaches in the context of 3D-TV applications, where such a utility distribution spans different layers. Finally, an additional requirement for the streaming applications based on UDP is to implement a *congestion-control* mechanism [10]. Its purpose is to dynamically adjust the transmission rate in response to the congestion state in the network, thus sharing the bandwidth fairly with concurrent applications and preserving the Internet stability.

Distributed delivery infrastructure. A network-distributed architecture has been explored in a number of commercial streaming Content Delivery Networks (CDNs) ([10], [7]). The CDNs consist of a large number (up to 10000) of geographically distributed servers, each of which can be configured to perform content caching, streaming, downloading or adaptation (e.g., transcoding). Alternatively, a number of proposals emerged for an overlay multicast infrastructure consisting solely of network end-hosts [7]. In this case, a multicast tree is formed employing the uplink bandwidth capacity of participating receivers, which self-organize into a distributed delivery architecture. Our main contribution is the first attempt to consider such architectures for implementing 3D-TV and FTV services. Our framework introduces virtual viewpoint creation as a new functionality to be implemented in a distributed manner. It is important to note that the feasibility of implementing FTV services within our framework is not constrained by the availability of distributed processing resources. Rather, if such resources are available, the framework leverages them to ease the burden on the server.

7 Conclusion

This paper presents a layered framework for 3D-TV transmission, combining multi-view 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). Important system aspects of our framework were extensively discussed and resulted in a few relevant conclusions. Firstly, the 3D-TV applications implemented within our framework can substantially benefit from adaptivity. Secondly, we discussed that particularly the image formation through warping (Section 3) introduces an interesting new QoS dimension for adaptivity. Thirdly, our proposed framework also offers an opportunity for selective reliability optimizations that cut across its different layers of information.

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] G. Petrovic and P.H.N. de With, "Near-future streaming framework for 3D-TV applications," in *to appear in International Conference on Multimedia and Expo (ICME)*, July 2006.
- [3] 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.
- [4] 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.
- [5] 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.
- [6] 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.
- [7] A. Ganjam and H. Zhang, "Internet multicast video delivery," *Proceedings of the IEEE*, vol. 93, no. 1, pp. 159–170, Jan. 2005.
- [8] S. Jin and A. Bestavros, "Cache-and-relay streaming media delivery for asynchronous clients," in *International Workshop on Networked Group Communication (NGC)*, 2002.
- [9] "Live555 Streaming Media," <http://www.live555.com/>.
- [10] G. J. Conklin, G. S. Greenbaum, K. O. Lillevold, A. F. Lippman, and Y. A. Reznik, "Video coding for streaming media delivery on the internet," *IEEE Trans. Circuits & Systems Video Technol.*, vol. 11, no. 3, pp. 269–281, Mar. 2001.
- [11] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *Intl J. Computer Vision*, vol. 47, no. 1-3, pp. 7–42, Apr. 2002.
- [12] D. Farin, "Automatic video segmentation employing object/camera modeling techniques," *Phd Thesis, Eindhoven University of Technology*, 2005.
- [13] Jonathan Walpole, Rainer Koster, Shanwei Cen, Crispin Cowan, David Maier, and Dylan McNamee, "A player for adaptive mpeg video streaming over the internet," in *Proc. SPIE Applied Imagery Patter Recognition Workshop AIPR-97*, Oct. 1997.
- [14] N. Feamster and H. Balakrishnan, "Packet loss recovery for streaming video," in *12th International Packet Video Workshop*, Apr. 2002.