OPTIMIZATION OF DISTRIBUTED FREE-VIEWPOINT VIDEO SYNTHESIS

Árpád Huszák

Department of Networked Systems and Services, Multimedia Networks and Services Laboratory
Budapest University of Technology and Economics, Hungary
huszak@hit.bme.hu

ABSTRACT

New type of video streaming may appear in the future networks, called Free-viewpoint Television (FTV) allowing customers to control their viewpoint and generate new views of a dynamic scene from any 3D position. The desired viewpoint is synthetized from two or more camera views requiring significant network and computational resources. We propose to distribute the viewpoint synthesis functionality in the network in so called proxy servers. In this paper we investigated how to localize the proxy servers in order to generate the lowest traffic load, but avoid the overload of the disposable resources. Finding the optimal layout is a NP-complete problem, so our goal was to analyze the achievable gain in a simple hierarchical network with several nodes, where the runtime of the brute force optimization algorithm is still acceptable. The obtained results paper showed that even 42% traffic decrease can be realized using distributed viewpoint synthesis for FTV services, so it is worth to continue our work and investigate new alternative optimization algorithms that are capable to find near optimal proxy server layouts and fast enough to adapt the proxy server arrangement continuously according to actual user requirements.

Index Terms — Free-viewpoint Television, streaming, distributed networks, viewpoint synthesis

1. INTRODUCTION

The free-viewpoint television is an interactive multimedia services offering similar functionalities that are known from 3D computer graphics. In contrast to 3D computer graphics applications, FTV targets real world scenes as captured by real cameras. Free-viewpoint streaming with its advanced features is foreseen as the next big step in 3D video technology beyond stereoscopy. FTV service allows users to switch between multiple streams to find views and viewing direction within a visual scene of their own choice. Slightly different viewing angles can be requested by the customers as they control their own viewpoint position and perspective, e.g. by moving or turning their head or changing position in a room. The customers of these interactive multimedia services may control the viewpoint and generate new virtual views of a dynamic scene. The uniquely generated and displayed views are composed from several high bitrate camera streams that must be delivered from the cameras to the viewpoint synthesis algorithm that can be deployed at the media server, at the client or in the network. The free-viewpoint video experience becomes more realistic as the number of camera views used to sample the scene increases, causing network traffic load increase, as well.

Two methods can be used to synthase an individual viewpoint from the camera sequences: Light Field Rendering (LFR) [1] and Depth Image Based Rendering (DIBR) [2]. The LFR algorithm interpolates a virtual view from multi-camera images, while DIBR uses fewer images and a depth map to establish new views [3]. Although many efforts have been done to compress LFR and DIBR, transmitting issues have not been deeply researched.

Only a few works have been published discussing the multiview video delivery problem. One of these works [4] presents a LFR based and QoS aware free-viewpoint video streaming solution. The paper focuses on I-frame retransmission and jump frame techniques in the application layer, supporting different level of QoS based on RTP/RTCP. Authors of [5] proposed a streaming system for DIBR based free-viewpoint video over IP networks, where depth video, texture video and common video are transmitted in individual RTP/RTSP flows. Other advanced ideas for transmission, like multipath delivery, P2P or cloud-assisted techniques for multiview video streaming were previewed in [6] and [7].

The FTV streaming models can be categorized based on the location of the virtual viewpoint synthesis. The first category is the server-based model, where all the camera views and corresponding depth map sequences are handled by a media server that receives the desired viewpoint coordinates from the customers and syntheses unique virtual viewpoint stream for each user. In this case only unique free-viewpoint video streams must be delivered through the network, but the computational capacity of the media server may limit the scalability of this approach. The second solution is to deliver required camera streams and depth sequences to the clients to generate their own virtual views independently. In this approach the limited resource capacity problem of the centralized media server can be avoided, but huge network traffic must be delivered in the network. The third model is a distributed approach, where the viewpoint rendering is done in distributed locations in the network. The distributed model can avoid bandwidth and computational resource overloads and handle the user requests in a scalable way. In this paper we focus on the third approach.

The only paper we found in the literature regarding to distributed FTV services was authored by Petrovic et al. [8]. They proposed an end-to-end delivery model for 3D video applications, which leverages a distributed system architecture to reduce the bandwidth and processing cost at the server and the end-hosts. Their prototype implementation demonstrated that highly heterogeneous clients can coexist in the system, ranging from auto-stereoscopic 3D displays to resource-constrained devices.

In this paper the distributed viewpoint synthesis functionality was investigated from the optimal layout point of view. Our aim was to find the optimal arrangement of the distributed viewpoint synthesis processes by allowing network nodes to act as proxy servers with caching and viewpoint synthesis functionalities. Our aim was to minimize the traffic load of the FTV service without overloading the computational and storage resources of the network components.

We found that layout optimization is a NP-complete problem, so in the current phase of the work our goal was to analyze the achievable gain in a simple network, where the runtime of the brute force optimization algorithm is still acceptable. The obtained results will be used in our future work with the aim to propose alternative optimization algorithms that are capable to find near optimal proxy server layouts that decrease the traffic load in a FTV network.

The remainder of the paper is structured as follows. After the introduction and the short survey of related works Section 3 presents our model for the distributed free-viewpoint television architecture. In Section 4 simulation results of the optimization approach were evaluated, while Section 5 highlights the main points of the presented work and suggests directions for future work.

2. DISTRIBUTED VIEWPOINT SYNTHESIS

Free-viewpoint video and television services allow users to individually change the desired viewpoint of a video scene that is captured by several cameras from several position. In order to produce the requested viewpoint, the camera streams must be delivered to the viewpoint syntheser algorithm. Due to very high storage capacity and computational resource requirements, we propose to distribute the viewpoint synthesis process in the network.

The distributed scheme can solve the scalability problems and keep the traffic load as low as possible. Our aim was to find the optimal deployment locations of the distributed viewpoint synthesis processes in the network by allowing network nodes to act as proxy servers with caching and viewpoint synthesis functionalities. These proxy servers may share their resources for viewpoint synthesis, recoding and caching purposes. Therefore, the user must not connect directly to the media server, but may ask the most appropriate proxy server for a synthetized stream with the desired viewpoint. The proxy servers will gather the camera streams that are needed to serve the connected clients and originate the unique streams as illustrated in Figure 1. Of course, different objectives can be targeted to locate the proxy servers. In this work our objective was to keep the overall traffic load as low as possible.

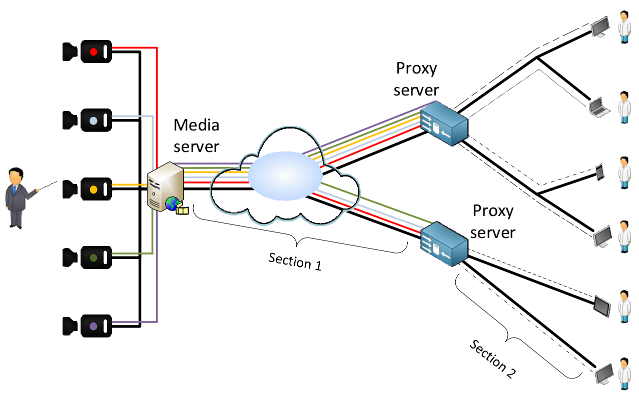

Figure 1. Proxy server based view synthesis

Basically, a proxy server may cache the segments of a conventional video stream, but in case of FTV services it is preferred to support codec functionalities and viewpoint synthesis, too. We modeled the proxy element as presented in Figure 2. The received camera streams (all or only a set of streams) are processed by the decoder module and stored in the cache. Based on the incoming viewpoint requests, the viewpoint synthesis module will generate the new stream that will be coded according to users' coding setups. The outputs of the proxy server are the unique, user specific streams.

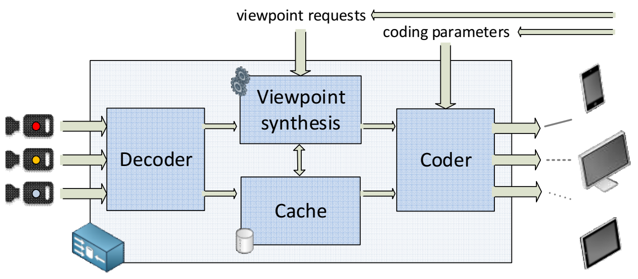

Figure 2. The caching and rendering proxy model

In this paper we analyzed how to localize the proxy servers in order to generate the lowest traffic load, but avoid the overload of computational and storage resources. In order to provide seamless viewpoint changes, the set of required camera stream must be available at the proxy server.

The path between the media server and each client can be divided into two parts. In section 1 (from media server to proxy server) the real camera streams are delivered, while section 2 (from proxy server to the client) the user specific view is transferred (Figure 1). The camera streams must be always forwarded with the highest quality and full resolution, requiring higher bandwidth, but fortunately multicast delivery mode can decrease the overall occupied bitrate on the links. In section 2 the streams are unique, so multicast is not an option. Opportunely, the synthetized streams may be coded with lower bitrate, e.g. if it is played out on a mobile terminal with low resolution display. By locating the viewpoint synthesis functionality closer to the camera sources, the high bitrate camera streams will occupy less total bandwidth in the network, but the proxy servers will have to serve more clients, so the total network traffic of the unique user specific streams will be higher. The goal is to determine the proxy locations to minimize the overall bandwidth:

$$\min\left\{\sum_{i=1}^{N}\left(Bw_i^{'}+Bw_i^{''}\right)\right\}, \qquad (1)$$

where N is the number of clients, $Bw_i^{'}$ and $Bw_i^{''}$ stands for the bitrate of the ith user in section 1 and section 2, respectively.

The described problem can be mapped to a knapsack combinatorial optimization task [9] if the *items* are the proxy servers, the *value* is the occupied total bandwidth (actually, it is the inverse of the bandwidth, because we want to minimize it and not maximize) and the limited *weight* is the computational and link capacity limit. Unfortunately the knapsack problem is known as NP-complete problem, so there are no guarantees that the optimal proxy topology setup can be found with acceptable runtime. In case of brute force method, all possible proxy location must be examined for each client. The complexity of the brute force optimization approach is $O(k2^n)$, where n stands for the number of possible proxy server locations and k is the number of users. Finding the optimal distribution of the viewpoint synthesis process is hard even in static network environment, where the clients do not change their point of access and the required camera streams necessary for the viewpoint production do not vary. In reality the problem is more difficult due to the continuously changing environment.

In this paper we analyzed the obtainable gain of controlled viewpoint synthesis process distribution in a FTV network. In the current state of the work a smaller network (spanning tree with several hierarchical levels) was examined. In this basic network the brute force was used to find the most appropriate topology layout of proxy servers in the network. Initially, all the clients require desired viewpoint and unique bitrate, determining two camera views used for the synthesis. The proposed brute force approach calculates the overall traffic on the network links for all proxy server setup combination and checks whether congestion occurs. Of course there are proxy server arrangements that are not acceptable because some of the links or computational resources of proxy servers may be overloaded. These setups must be discarded.

Although the brute force algorithm as not acceptable for complex networks and changing environment, it makes possible to estimate the achievable performance increase using distributed proxy servers. The next step of our work will be to find real time optimization methods, however it is clear that faster proxy serves topology optimization process will not be able to provide the globally optimal arrangement for this NP-complete problem.

3. SIMULATION RESULTS

The distributed viewpoint synthesis was analyzed in a simulation tool implemented in Java. The tool provides an interface to design the FTV network topology, set link capacities, add users and their requested viewpoints, determine the camera set and define network elements to act as router or proxy server. Proxy server (viewpoint synthesis) functionality can be added manually to a network element, or using the optimization process that locates the proxy servers based on the optimization process. In the current development state brute force approach was used. The advantage of the brute force method is that it returns the global optima, but it works only for small network (cca. 20 network elements and 60 clients) due to long runtime. We used a hierarchical network topology with two to four levels with 3-5 clients attached to each lowest level network element. The default values of link capacities were set to 100 Mbps, the camera stream bitrates were 10 Mbps, while the individual virtual view stream bitrates were set between 3 and 10 Mbps.

The brute force algorithm compares all proxy setup combination and checks whether it is a correct topology arrangement or not. While the network links capacities are limited, in some of the network layouts the links can be overloaded. These proxy server arrangements must be dropped. The number of total layouts increases exponentially, similarly as the optimization process runtime. Table 1 shows the number of total and acceptable proxy server arrangements in a two, three and four level hierarchical network. While the distributed proxy server optimization algorithm ran fast for two and three levels, it took 66 hours for a four level network, with 21 network elements and 60 clients.

Table 1. Number of proxy server layouts

Hierarchical levels	Total layouts	Correct layouts	Duration [ms]
2	8	1	1
3	512	124	190
4	2 097 152	704 969	239 927 390

In order to synthetize a desired viewpoint video according to the user's request, at least two camera streams are used. Usually the two requested cameras are capturing the scene from close positions, so the synthetized virtual view will provide better visual quality than using the streams of two cameras located farther from each other. In the first scenario we analyzed how the camera stream bitrate effects the total occupied bandwidth on the network. The camera bitrate has impact on the first section of the delivery path (media server – proxy server) as discussed in the previous section (Figure 1). We changed the camera stream bitrate from 5 Mbps to 25 Mbps while keeping other parameters constant. The total traffic in three different proxy server distribution layouts was measured in a network with three levels. In Figure 3 the layout #7 is the optimal, while layout #511 is the worst proxy server setup in case of 25 Mbps camera stream bitrates. Layout #311 has been selected as an arrangement with average performance. We compared only the above mentioned three layouts, however in case of other camera stream bitrates the optimal proxy server setup differs.

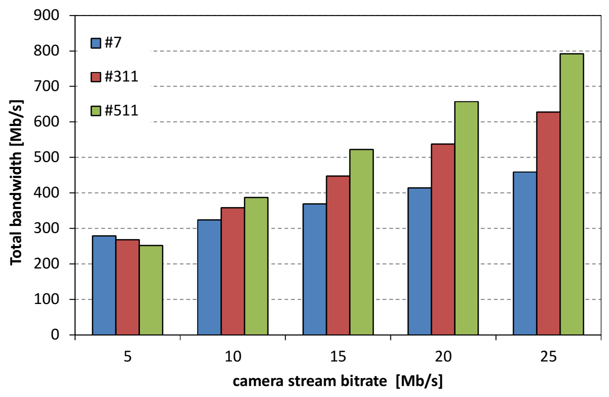

Figure 3. Comparison of three layouts

The results show that the optimally distributed viewpoint synthesis functionality decreases the total traffic by 42% compared to the worst case scenario when the camera bitrate is set to 25 Mbps. The improvement is 21% compared to layout #311. When the camera bitrate is not 25 Mbps, layout #7 is not the optimal proxy server setup, but it can be observed that performs better than layout #311 and #511 in most of the cases.

To evaluate the performance of the distributed viewpoint synthesis, we compared the total network bandwidth of all proxy server topology setups. Of course, those layouts were discarded, where link overload occurred. We have compared a two network topologies with three and four hierarchical levels. In the case of the first one 10 network elements and 18 clients were present, while in the second case 21 elements with 60 attached clients were in the network. All the network elements, except media server, were able to serve as a proxy server with viewpoint synthesis functionality. Due to presentation reasons, the analyzed layouts were ordered by the measured total network traffic and illustrated in Figure 4. The results are shown in a single figure, hence the overall bandwidth values of the four level network topology are on the left vertical axis of the figure, while the right vertical axis is scaled for the three level network topology. There is also difference in the horizontal axis scale.

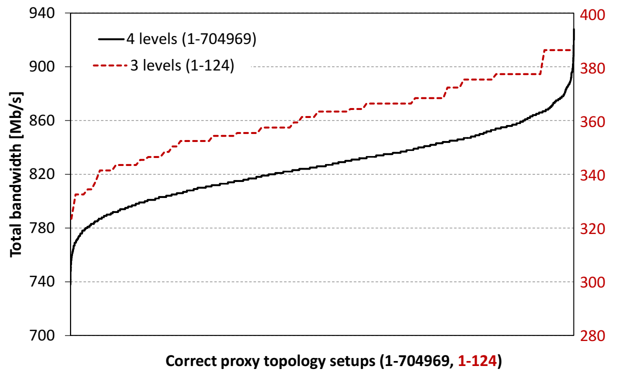

Figure 4. Comparison of correct layouts in a three and four level topology

The simulation results make it possible to compare the occupied bandwidth of all proxy server layouts in a fixed network structure. As Figure 4 shows there are significant differences in performance. In case of a three level network the overall traffic can be reduced by 12%, while in a four level network the gain is 16% compared to the worst case scenario. The results prove that it is worth to consider viewpoint synthesis distribution for free-viewpoint television service.

The other important constituents of a free-viewpoint video streaming service are the clients. Therefore, we analyzed the performance variation in function of the number of clients. A three level network hierarchy was assumed with 10 network element that can be simple router or act s proxy server, except media server. In this environment 512 different layout combinations exist, but only 124 are correct that means no link overload occurs. The total network traffic in in case of 18 and 60 clients are presented in Figure 5. The proxy server layout combinations are ordered by bandwidth values.

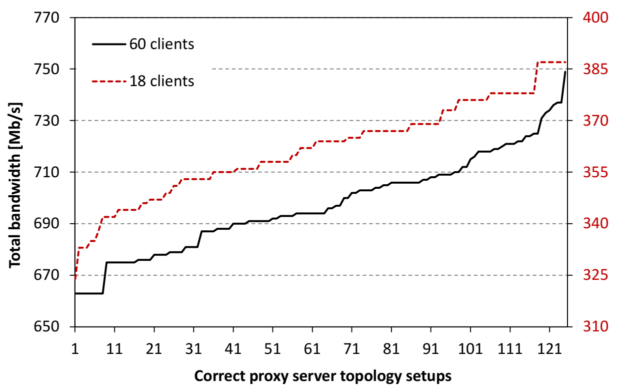

Figure 5. Comparison of layout performance in function of client density

According to the obtained simulation results, the difference between the worst layout and the optimal one is more than 10% in both cases. However, by increasing the number of clients the overall network traffic also increases, but the relation ratio is not the same. Comparing the occupied network bandwidth of the optimal viewpoint synthesis distribution arrangements, we can find that having 3.3 times more clients result only 2.1 times higher traffic load. The reason of this behavior is that due to high number of client, all camera streams are already forwarded through section 1 of the delivery path (Figure 1). By increasing the number of clients, the traffic load on section 1 will not increase, only in section 2.

4. CONCLUSIONS

Free-viewpoint television is a promising approach to offer freedom for users while watching multiview video streams. Each user viewpoint is synthetized from at least two high bitrate camera streams that are used to capture the scene. Both stream delivery and viewpoint generation are resource hungry processes leading to scalability issues in a complex network. In this paper the distributed viewpoint synthesis approach was analyzed in order to offer scalable solution for FTV services. We defined a network element with viewpoint synthesis and transcoding capability, called proxy server. Our aim was to propose an optimal layout of proxy servers in order to minimize the overall traffic in the network but avoiding link congestions. The optimization problem is similar to the knapsack combinatorial optimization task that is proved to be a NP-complete problem, hence we analyzed the achievable gain in a simple hierarchical network with several nodes, where the runtime of the brute force optimization algorithm is still acceptable. In this phase of the work our goal was to estimate the traffic load decrease in case of distributed viewpoint generation. Our future plans are to propose alternative real-time optimization algorithms that are capable to find near optimal proxy server layouts in a continuously changing environment in order to decrease the traffic load in the FTV network. The new algorithms must be fast enough to adapt the proxy server arrangement continuously according to actual user requirements. Our results presented in this paper showed that significant traffic decrease can be realized using distributed viewpoint synthesis for FTV services.

ACKNOWLEDGEMENT

The research leading to these results has received funding from the European Union's Seventh Framework Programme ([FP7/2007-2013]) under grant agreement n° 288502 (Concerto project). This work was also supported by the TÁMOP-4.2.2.C-11/1/KONV-2012-0001 project. The authors are grateful to the many individuals whose work made this research possible.

REFERENCES

[1] M.Levoy and P.Hanrahan., "Light field rendering", Computer Graphics, *Proceedings. SIGGRAPH96*, August 1996

[2] Christoph Fehn, "Depth-image-based rendering (DIBR), compression, and transmission for a new approach on 3D-TV", *Proc. of SPIE*, *Vol. 5291, Stereoscopic Displays and Virtual Reality Systems*, May 2004, pp. 93-104

[3] Masayuki Tanimoto, Mehrdad Panahpour Tehrani, Toshiaki Fujii, Tomohiro Yendo, "Free-Viewpoint TV". *IEEE Signal Process. Mag*. 28(1): 67-76, 2011

[4] Zhun Han; Qionghai Dai, "A New Scalable Free Viewpoint Video Streaming System Over IP Network," *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2007. ICASSP 2007., vol.2, no., pp.II-773,II-776, 15-20 April 2007

[5] Goran Petrovic and Peter H. N. de With, "Near-future Streaming Framework for 3D-TV Applications", *ICME2006*

[6] Chakareski, J., "Adaptive multiview video streaming: challenges and opportunities", *Communications Magazine, IEEE* , vol.51, no.5, pp.94,100, May 2013

[7] Gurler, C.G., Tekalp, M., "Peer-to-peer system design for adaptive 3D video streaming," *Communications Magazine, IEEE* , vol.51, no.5, pp.108,114, May 2013

[8] G. Petrovic and D. Farin, "A distributed delivery model for 3D-video streams." *Proceedings of the First International Conference on Immersive Telecommunications*, ICST, 2007.

[9] H. Kellerer, U. Pferschy, and D. Pisinger. Knapsack Problems. *Springer*, 2004