Predictive Multicast Group Management for Free Viewpoint Video Streaming

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

***Abstract*—Free viewpoint video (FVV) is a new approach of interactive streaming services, where users are able to freely change their viewpoint. The desired viewpoint is synthetized from two or more camera views that must be delivered to the users depending on their continuously changing perspective. Multicast delivery of camera streams is an appropriate solution, however due to network latency and frequent changes of the viewpoint, the required camera streams may arrive too late, interrupting the FVV synthesis and playout. In this paper a seamless FVV streaming scheme is presented based on user viewpoint prediction. In order to avoid the starving of the FVV synthesizer, we propose to use threshold areas to prefetch the camera views that will be probably required for the viewpoint generation. We have formulated the calculation steps of the threshold values in order to minimize the starvation ratio and its duration. The obtained simulation results show that using the predictive multicast management scheme, the clients receive the required camera views in time in more than 95% of the cases. Moreover, we showed that the number of used FVV cameras and the number of clients have significant impact on the performance of the FVV service.**

Keywords—Free Viewpoint Video; multicast; streaming;

I. INTRODUCTION

Free viewpoint video (FVV) offers similar functionalities that are known from 3D computer graphics. FVV service allows users to choose own viewpoint, viewing direction and interactive free navigation within a visual scene. In contrast to 3D computer graphics applications, FVV targets real world scenes, captured by real cameras.

FVV streaming is foreseen as the next big step in 3D video technology beyond stereoscopy. However, a commercial FVV service will be similar to the IPTV solutions, the difference is that multiview video is required to provide free view functionality, which enables viewers to see a 3D scene from slightly different viewing angles as they control their own viewpoint position and perspective, e.g. by moving or turning their head. The free viewpoint video experience becomes more realistic as the number of camera views used to sample the viewing cone increases. Therefore, the network bandwidth required to transmit multiple camera views for the viewpoint synthesizer deployed in the user equipment can overload the network capacity.

Although many efforts have been done to compress Light Field Rendering (LFR) [1][2] and Depth Image Based Rendering (DIBR) [3], transmitting issues have not been deeply researched.

Delivery of FVV is different from traditional video streaming in the following points. Firstly, FVV requires several video streams captured by different cameras recording the scene from different locations, hence synchronization in streaming process among all cameras must be done. The second point is that the camera streams required by customers may change frequently because of free navigation of viewpoint, so variation of visual quality due to view switching must be handled. Thirdly, FVV streaming costs more bandwidth than single video stream, therefore scalable quality of service is an important issue.

IP multicasting is suited for both video on demand as well as live multimedia applications. In case of FVV multicast delivery, streams of camera views are transported over separate IP multicast groups, so that all users can selectively join the multicast groups that are required to synthetize the desired viewpoint. Multicast transmission is effective to reduce the network load, but continuous and frequent viewpoint changes may lead to interrupted FVV service due multicast group join latencies. The required camera streams may arrive too late and starve the FVV synthesizer process.

In this paper multicast FVV scheme is proposed based on user viewpoint prediction. To prevent the user's viewpoint synthesizer algorithm from remaining without camera streams, multicast group join threshold is introduced in order to provide all camera views that may be requested in the near future. In order to find the optimal threshold values, the multicast groups join latency and viewpoint movement features were examined. The performance of the prediction based adaptive threshold setup was analyzed in Ns-2 simulations.

The rest of this paper is organized as follows. The background of free viewpoint video streaming systems and viewpoint synthesis methods are presented in Section II. In Section III, the proposed predictive multicast group management scheme for FVV services is introduced. The obtained performance results are presented in Section IV. The summary of the paper and the conclusions can be found in the last section.

978-1-4799-3200-9/14/$31.00 ©2014 IEEE

II. Background of Free Viewpoint Video Streaming

Media content delivery requires high link capacity and low latency in order to provide acceptable quality of media streams. The transmission of traditional high resolution single-view videos is still challenging, but in case of multi-view videos this challenge becomes more interesting.

To synthetize a virtual viewpoint from existing camera views, the camera streams must be forwarded to the renderer that can be deployed in the user equipment, in a media server, or distributed in the network. Without compression, the delivery of camera stream set would be impossible. An efficient way to encode two or more videos showing the same scenery from different viewpoints is known as multi-view video coding (MVC) [4][5]. MVC is an extension of H.264/AVC that exploits both inter-view and temporal redundancies for efficient compression and keeps full resolution of all views.

To generate an individual viewpoint from the camera sequences two methods can be used: Light Field Rendering (LFR) [1][2] and Depth Image Based Rendering (DIBR) [3]. LFR interpolates a virtual view from multicamera images, while DIBR uses fewer images and a depth map to establish new views [6]. A depth map is an image in which the intensity of each pixel represents distance between the camera and the surface of an object as illustrates in Fig. 1 .

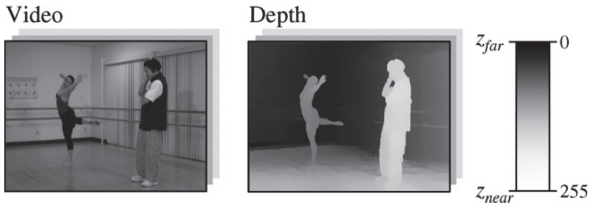

Fig. 1. Video frame with depth map

Continuous depth data (captured via depth or infrared cameras) is very important in 3D warping algorithms for high quality virtual image interpolation. The standard that supports video plus depth is known as MPEG-C Part 3 [7]. In case of DIBR at least two camera streams and the corresponding depth map sequences must be available at the renderer to generate an individual viewpoint [8] (Fig. 2).

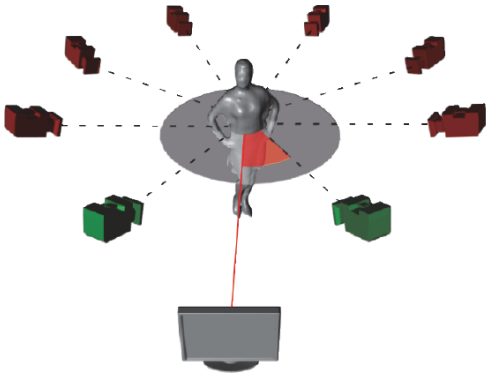

Fig. 2. View synthesis

In the first FVV solutions offline viewpoint generation was mainly used in film production, e.g. for stop-motion special effects in movies or for sports effects systems, like "LiberoVision" [9]. Fortunately, the increased computational and network resources makes interactive real-time FVV services available, too.

Although compression of LFR- and DIBR-based FVV was studied intensively, transmitting issues have not been deeply researched. Only a few works in research literature have discussed the multi-view video delivery problem. One of these works is in [10] presenting a LFR based and QoS aware FVV streaming solution. The paper focuses on I-frame retransmission and jump frame techniques in the application layer based on RTP/RTCP to support different level of QoS. Authors of [11] introduced a streaming system for DIBR based FVV over IP networks. They proposed to divide video streams into depth video, texture video and common video, and transmit them in RTP/RTSP individually, but did not solve view switching and synchronization problems.

Selective streaming is a method to reduce the bandwidth requirements of multi-view video, where only a subset of views is streamed depending on the user's viewing angle. To select which views should be delivered, the viewer's current head position is tracked and a prediction of future head positions is calculated [12]. Kurutepe et al. [13] presented a multi-view streaming framework using separate RTSP sessions to deliver camera views. The client may choose to initiate only the required number of sessions. The proposed scheme utilizes currently available streaming protocols with minor modifications. Multicast FVV transport solutions were neither investigated deeply. Authors of [14] proposed a multi-view video streaming system based on IP multicast. The multi-view videos are transmitted using multiple-channel scheme to support various users who have different available bandwidth. Other advanced ideas for transmission, like multipath delivery, P2P or cloud-assisted techniques for multiview video streaming were reviewed in [15].

Three different FVV streaming models can be distinguished regarding to virtual viewpoint synthesis. In the first server-based model all the camera views and corresponding depth map sequences are handled by a server that receives the desired viewpoint coordinates from the customers and syntheses unique FVV stream for each user. In this case only unique FVV 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 the required camera views and depth sequences to the clients that generate their own FVV stream independently. In this client-based approach at least two camera and depth views must be forwarded to each client to perform the viewpoint synthesis. In this approach the limited resource capacity problem of the centralized FVV media server can be avoided, but huge amount of camera streams must be delivered through the network. The third model is a distributed approach, where the viewpoint rendering is done in distributed locations in the network. In this work we focus on the second client-based model and propose a predictive solution for multicast group management in order to provide seamless viewpoint changes.

III. PREDICTIVE MULTICAST GROUP MANAGEMENT FOR FVV SERVICES

Due to the huge data amounts transferred through the network, multi-view video's delivery remains challenging. Fortunately, multicast delivery may be a solution to reduce the required FVV service bandwidth. In case of multicast free viewpoint video streaming each camera view is encoded and forwarded on a separate channel to the users. The separate channels (camera views) can be accessed by joining the multicast group that contains the needed camera source, as illustrated in Fig. 3. Users can switch views by subscribing to another multicast channel, while leaving their present one. Conceptually, the operation of this system is analogous to that of IPTV.

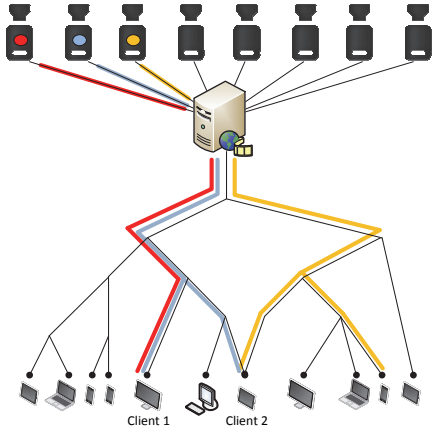

Fig. 3. Multicast FVV

Multicast is effective to reduce the network load, but continuous and frequent viewpoint changes may lead to interrupted FVV service due multicast group management message latencies. To generate a desired virtual perspective, the client must be joined to at least two multicast groups that contain the required camera views. When the viewpoint is changing and new camera views are needed, the client must join to a new multicast group ensuring the actually needed camera stream. If the multicast group change (leaving the old multicast group and joining the new one) is performed only when the new virtual view already must have appeared on the screen, there will be an interrupt in the FVV stream, because the lately requested camera view stream will not be received in time to synthetize the new viewpoint. Therefore, our aim was to propose a viewpoint prediction based solution for camera view handoffs to minimize the probability of the synthesis process starving.

To prevent the user's viewpoint renderer algorithm from remaining without camera stream source, multicast group join threshold can be introduced in order to provide all camera streams that may be requested in the near future. In the illustrated scenario (Fig. 4) the cameras are deployed in line and the user can change the required viewpoint within a fixed width zone determined by the line of the cameras.

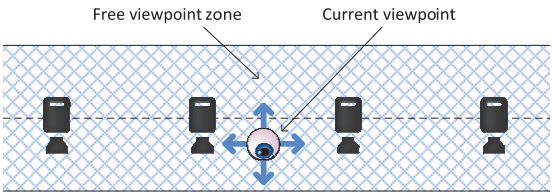

Fig. 4. Free viewpont zone

In the scenario, depicted in Fig. 5, the viewpoint of a user is freely changing within the zone. Using the proposed viewpoint prediction model and supposing that the viewpoint of the client is moving from the *blue* camera view position towards the *yellow* one in this sample scenario, the desired view will reach *Threshold*_1 initiating a multicast joint message to the *yellow* camera stream group. While the viewpoint of the client is within the threshold zone, it will become a member of three multicast groups (*blue*, *green* and *yellow*). If the viewpoint is moving towards the yellow camera position and reaches the *Threshold*_2, the client should leave the *blue* multicast group.

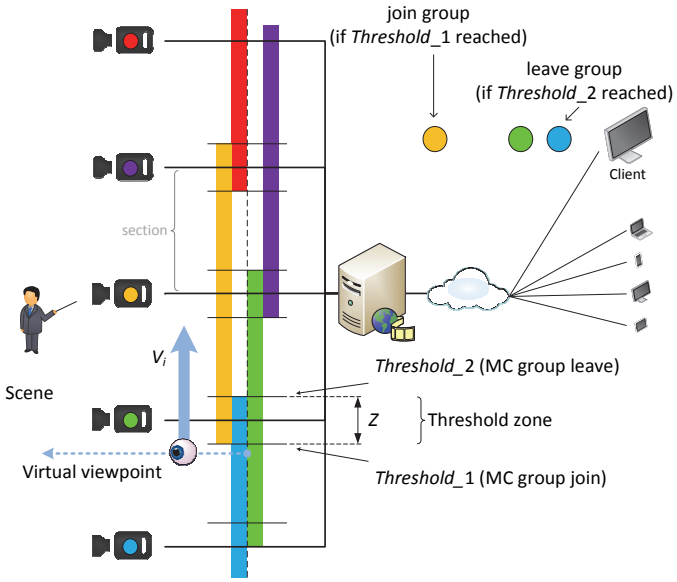

Fig. 5. Multicast FVV: Multicast group join thresholds

However, Fig. 5 shows a linear camera setup (1 dimensional camera topology), the cameras can be deployed in plane (2D) as well as in space (3D). In the latter cases not only two camera streams are required for the viewpoint synthesis, but three or even four that makes the threshold area determination more difficult. Our goal was to keep the threshold area as low as possible to reduce the number of multicast group memberships, so the overall network bandwidth, but keep it large enough to avoid playout interruption during viewpoint changes.

In order to find the optimal threshold values, the multicast groups join latency and viewpoint movement features must be considered. Assuming a linear camera row (one dimensional topology), where x_i denotes the actual viewpoint position and v_i the velocity of the viewpoint in time t_i, the next viewpoint location in time t_{i+1} can be expressed as follows

$$x_{i+1} = x_i + v_{i+1}\left(t_{i+1} - t_i\right). \quad (1)$$

Depending on the velocity of the viewpoint in the next moment (v_{i+1}), the view synthesis algorithm may require other camera views than in the previous moment. The problem is that v_{i+1} is not known in t_i time, so it must be estimated based on previous viewpoint movement behavior. We used linear regression method to estimate the next viewpoint by calculating the average viewpoint velocity values from previous viewpoint coordinates.

To determinate the threshold values and zones of the viewpoint coordinates that triggers the multicast join and leave processes, the required time duration (d_m) from sending a multicast join message to receiving the first packet of the camera stream is necessary. The client can only decode the multicast stream after receiving an I-frame, therefore the I-frame period time (d_I) must be also taken into consideration. Within $D=d_m+d_I$ time the viewpoint location should not move to another section of the camera row, where new camera streams are required for the viewpoint synthesis. In our proposed method the threshold zone dimensions (Z) is determined as follows (see Fig. 5)

$$Z \geq 2\left(v_{i+1} \cdot D\right), \quad (2)$$

where D is assumed to be the sum of d_m (RTT (round-trip time) between the client and the FVV media server) and d_I (time distance between the I-frames), while v_{i+1} is estimated as

$$v_{i+1} = \frac{\sum_{j=1}^{i} v_j}{i}. \quad (3)$$

In some cases the d_m parameter can be even lower than the RTT if the join message goes through a multicast router that is close to the client that already forwards the required camera stream to other users. If the camera view must be inquired from the media server, the multicast join latency will be equal to RTT. In order to minimize the viewpoint synthesis algorithm starvation, we used $\max\left(d_m\right) = RTT$ in our model. According to Fig. 5 the threshold zone size can be calculated also as

$$Z = Threshold\_2 - Threshold\_1. \quad (4)$$

To avoid camera view starving, the new camera stream must be prefetched when the current viewpoint enters the threshold zone. The threshold values in each section can be determined based on the camera coordinates (c_k) and the threshold zone size (Z) as $c_k \pm Z/2$. In the forthcoming evaluation section, $w=Z/2$ was used as a parameter, named window size.

From architectural point of view, the proposed solution will require multicast support in the network layer. The generally used PIM-SM [16] or PIM-DM [17] protocols are applicable for the presented free viewpoint video streaming service without any modification. Using PIM-SM rendezvous point (RP) and routers with multicast support are necessary elements of the network, while the control of group management packages must be done in the application layer. Synchronization of camera streams are also required in order to perform seamless camera handovers. Using the RTP/UDP timestamp feature this problem can be handled.

IV. Simulation Results

In order to test the performance of the proposed predictive multicast FVV streaming model described in the previous section, we analyzed some scenarios with Ns-2 [18] network simulator. In the simulated network topology the routers were deployed in three hierarchical layers as illustrated in Fig. 6.

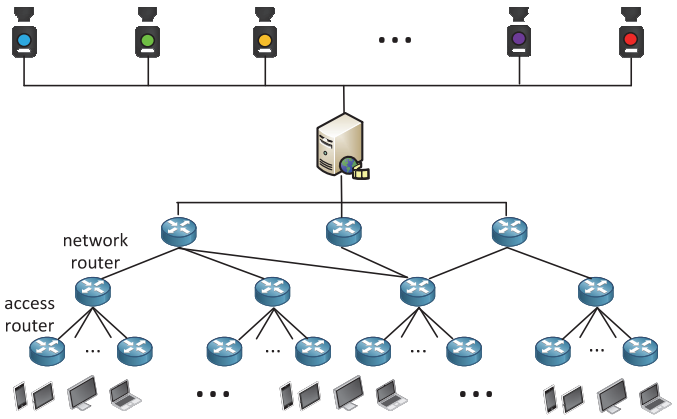

Fig. 6. Simulatied FVV network topology

The simulation environment made it possible to set the number of cameras (equal to number of multicast groups) and users, link characteristics, viewpoint movement behavior, camera stream bitrates, etc. The default parameter values used for the examination of the predictive FVV streaming model are presented in Table I. In the deployed FVV streaming simulations PIM-DM multicast dense mode protocol was used.

TABLE I. Default parameter values

Parameter	Default value
simulation time	20 s
link delay	10 ms
total number of access routers	50
total number of clients	350
number of cameras	25
camera stream GOP size	1
packet size	1000 byte
video bitrate per camera	1 Mbps
link bandwidth	150 Mbps
viewpoint velocity (v)	avg. 0.3 × camera distance per timeslot
max. timeslot length	avg. 0.05 sec, rand(0; 0.1)
window size ($w=Z/2$)	0.3 × camera distance

In the evaluation phase, the viewpoint velocity was considered to be measured in camera distance unit (the distance between two neighbor camera positions is *cam_dist*=1) per timeslot, where the timeslot is a random variable. In other words, the viewpoint shifts with viewpoint velocity value in random times. The time difference between two viewpoint shifts can be set with the max. timeslot length parameter.

In the first simulation scenario we analyzed the correlation of the average viewpoint velocity values and the window size. The velocity parameter was changed from 0.2 to 1 camera distance units. When the velocity parameter is 1, the viewpoint will skip always to a new section in the camera row and require new camera streams (Fig. 5). The threshold zone width is adjusted by the window (w) parameter. The w=0.5 setup means that the viewpoint is always in a threshold zone, thus the client is requesting three camera views continuously. However, there is no guarantee that required camera stream is received in time and no starvation occurs as Fig. 7 shows. If the viewpoint velocity is too high or the network latency increases, the required camera streams will not be available at the client.

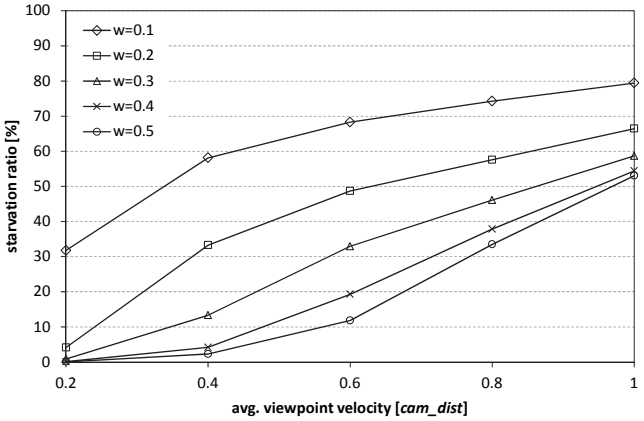

Fig. 7. Starvation ratio in function of viewpoint velocity and window size

According to the results, if the window size (w) is set independently from the viewpoint velocity (v), the starvation ratio can be very high. However, it is important to note that the quality degradation effect of the starvation significantly depends on its duration. Using the proposed scheme for the window size setup, the starvation ratio can be kept low. E.g., when v=0.2 and v=0.4, the RTT is 60 ms and the timeslot between the viewpoint shifts is 0.05 s, the proposed window size according to (2) is w=0.24 and w=0.48, respectively. Utilizing the proposed scheme the starvation ratio is below 3% in both cases.

The appropriate window size (threshold zone) setup can minimize the starvation effect as analyzed in the next simulation scenario. The comparison of viewpoint velocity values and the caused starvation ratios are presented in Fig. 8. Based on the obtained measurement results, if the window size is set based on the velocity of the viewpoint the synthesizer algorithm will get the camera views in time in more than 95% of the cases as the results shown in Table 2. While setting the threshold zone too narrow, the starvation ratio can reach even 57%.

TABLE II. STARVATION RATIO IN CASE OF THE PROPOSED SCHEME

viewpoint velocity (v)	proposed window size (w)	starvation ratio
0.1	0.12	3.93%
0.2	0.24	4.45%
0.3	0.36	4.22%
0.4	0.48	4.42%

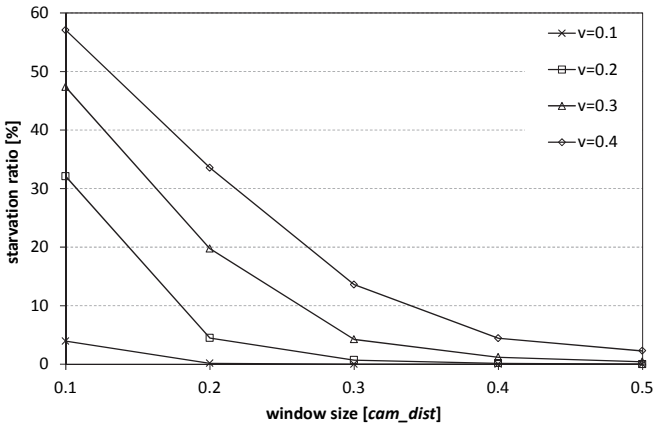

Fig. 8. Starvation ratio

The number of cameras in the FVV system can be very high in order to provide high quality synthetized viewpoint videos. By increasing the number of deployed cameras, the required streams will become more unique and there will be more camera streams that are not requested at all, or only few users are joined to a specific camera multicast group. Hence, the multicast join latency will increase, because the probability that a router in the lower topological level already receives the requested stream is lower, so join message and the video packets will travel on longer path. We have measured how the number of FVV cameras affects the starvation. The obtained results are shown in Fig. 9.

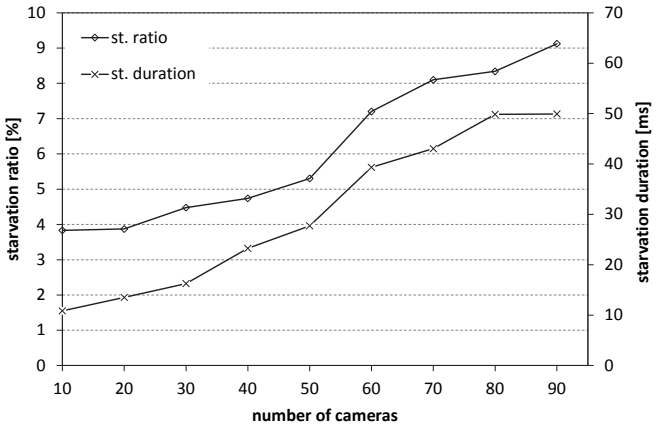

Fig. 9. Starvation ratio and duration

We found that as the starvation ratio as the duration is significantly higher if more cameras are used. If only 10 cameras are deployed, ca. 35 users are joined to each camera multicast group, while using 90 cameras this number is only 3.9 in average. The number of customers in the multicast group has significant impact on the starvation ratio and its duration, it can even multiply these values.

The number of users is the other parameter that influences the number of multicast group members of a camera view. If the FVV system serves more customers, the multicast groups will contain more users, so the latency of camera stream reception can be decreased. The reason is the same as it was

described in the previous scenario. Namely, the routers in the FVV network already forwards the desired multicast views to other clients with higher probability. The measurement results are presented in Fig. 10.

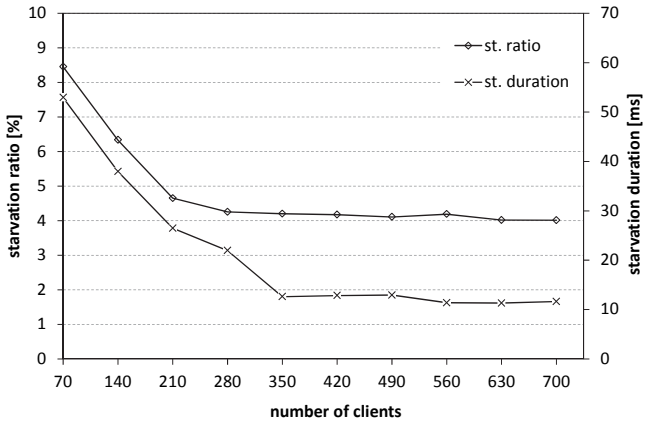

Fig. 10. Client number impact on starvation ratio and duration

Low number of users significantly decreases the performance of the FVV service. However, in the simulated environment neither the starvation ratio nor its duration changed when the number of clients was higher than 300. The reason is that all the access routers already forwards all camera streams, thus the requested content is only in one hop distance from the client.

V. CONCLUSIONS

Free viewpoint video is a promising approach to offer freedom for users while watching multiview video streams. The delivery of camera views required for viewpoint synthesis can overload the network without multicast streaming, however late multicast group joins may lead to starvation of the FVV renderer process. In this paper we proposed a prediction based multicast group management scheme to avoid late camera view delivery. The introduced solution uses threshold areas, where instead of the two necessary camera streams, three views are forwarded at camera section borders. We have formulated how to calculate the threshold area in order to minimize the starvation ratio and its duration. The proposed threshold calculation depends on the viewpoint velocity and the network delay. According to the obtained results the viewpoint synthesizer algorithm gets the camera views in time in more than 95% of the cases if the proposed scheme based on the velocity of the viewpoint and the RTT is used. We also observed that the starvation ratio and its duration highly depend on the number of deployed cameras and customer density. FVV streaming is a new form of 3D media delivery that was not intensively investigated before. Hopefully, FVV streaming will become a popular interactive multimedia service of the near future.

ACKNOWLEDGMENT

The research leading to these results was supported by the European Union, co-financed by the European Union's Seventh Framework Programme ([FP7/2007-2013]) under grant agreement n° 288502 (CONCERTO project) and 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] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen, "The lumigraph", Computer Graphics, SIGGRAPH-96, Aug.1996

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

[4] K. Mueller, P. Merkle, A. Smolic, and T.Wiegand, "Multiview coding using AVC," MPEG2006/m12945, 75th MPEG meeting, Bangkok, Thailand, Jan. 2006

[5] P. Merkle, A. Smolic, K. Mueller, T. Wiegand, "Efficient prediction structures for multiview video coding", IEEE Transactions on Circuits and Systems for Video Technology, Special Issue on Multiview Video Coding and 3DTV, 2007

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

[7] ISO/IEC JTC 1/SC 29/WG 11. Committee Draft of ISO/IEC 23002-3 Auxiliary Video Data Representations. WG 11 Doc. N8038. Montreux, Switzerland, April 2006.

[8] J. Starck, J. Kilner, and A. Hilton. A Free-Viewpoint Video Renderer. Journal of Graphics, GPU, and Game Tools, 14(3):57--72, Jan. 2009.

[9] www.liberovision.com

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

[11] Goran Petrovicand Peter H. N. de With, "Near-future Streaming Framework for 3D-TV Applications", ICME2006

[12] Gürler, C.G.; Görkemli, B.; Saygili, G.; Tekalp, A.M., "Flexible Transport of 3-D Video Over Networks," Proceedings of the IEEE , vol.99, no.4, pp.694,707, April 2011

[13] E. Kurutepe, A. Aksay, C. Bilen, C. G. Gurler, T. Sikora, G. B. Akar, and A. M. Tekalp, "A standards-based, flexible, end-to-end multi-view video streaming architecture", in Proc. Int. Packet Video Workshop, Lausanne, Switzerland, Nov. 2007, pp. 302–307.

[14] Li Zuo; Jian Guang Lou; Hua Cai; Jiang Li, "Multicast of Real-Time Multi-View Video," Multimedia and Expo, 2006 IEEE International Conference on , vol., no., pp.1225,1228, 9-12 July 2006

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

[16] Fenner B. et al., Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification, RFC 4601, August 2006

[17] A. Adams, J. Nicholas, W. Siadak, Protocol Independent Multicast - Dense Mode (PIM-DM), RFC 3973, January 2005

[18] Ns-2 – Network Simulator, http:///www.isi.edu/nsnam/ns/index.html