Scaling virtual camera services to a large number of users

Vamsidhar Reddy Gaddam[1,*], Ragnar Langseth[1], Håkon Kvale Stensland[1],
Carsten Griwodz[1], Dag Johansen[2], Pål Halvorsen[1]

[1]Simula Research Laboratory & University of Oslo, Norway
[2]University of Tromsø, Norway

[*]vamsidhg@ifi.uio.no

ABSTRACT

By processing video footage from a camera array, one can easily make wide-field-of-view panorama videos. From the single panorama video, one can further generate multiple virtual cameras supporting personalized views to a large number of users based on only the few physical cameras in the array. However, giving personalized services to large numbers of users potentially introduces both bandwidth and processing bottlenecks, depending on where the virtual camera is processed.

In this demonstration, we present a system that address the large cost of transmitting entire panorama video to the end-user where the user creates the virtual views on the client device. Our approach is to divide the panorama into tiles, each encoded in multiple qualities. Then, the panorama video tiles are retrieved by the client in a quality (and thus bit rate) depending on where the virtual camera is pointing, i.e., the video quality of the tile changes dynamically according to the user interaction. Our initial experiments indicate that there is a large potential of saving bandwidth on the cost of trading quality of in areas of the panorama frame not used for the extraction of the virtual view.

Categories and Subject Descriptors

H.5.1 [**Multimedia Information Systems**]: Video; I.4.9 [**Applications**]: Video

General Terms

Experimentation; measurement; performance

Keywords

Interactive immersion; panorama video; zoom, panning; real-time; virtual camera, video streaming

ACM MMSys'15, Portland, Oregon, USA
Copyright 2015 ACM xxx-x-xxxx-xxxx-x/xx/xx
http://dx.doi.org/xxxxxx/xxxx ...$15.00.

1. INTRODUCTION

There exist many types of panorama solutions where high-resolution, wide field-of-view video is captured and streamed in real-time. For example, in arena sports like soccer, American football and ice-hockey, many game analysis systems provide camera arrays where individual camera images are stitched together to cover the entire field. Then, to focus on parts of the area, it is often desirable to zoom and pan into the generated video. Figure 1 demonstrates an example output of such a system. In this case, a virtual camera is generated by extracting pixels from parts of the stitched panorama video allowing individual users to interactively control an own personalized view. However, these types of systems also give interesting opportunities for innovation in broadcasting scenarios where large number of fans and supporters would like to generate their own camera view of the event.

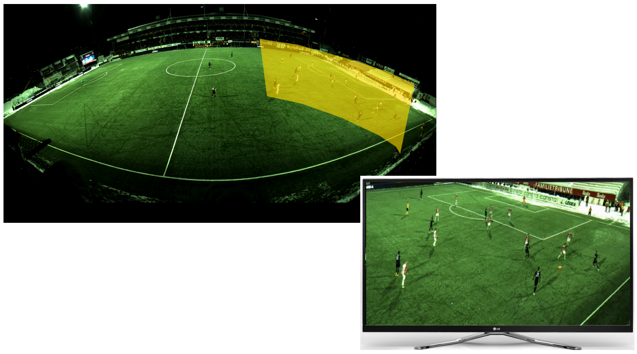

Figure 1: The panorama video with the marked region of interest is shown together with the generated virtual camera, emphasizing that the extracted area is not a simple crop from the high-resolution panorama video. It is generated by a perspective reprojection and hence we cannot achieve it by a simple rectangular cropping.

We have earlier described the Bagadus system [8, 15] generating panorama videos of a soccer stadium in real-time. Additionally, we have presented how individual users can be his own cameraman [5, 6] by extracting a zoomed and panned video from the panorama video, for example following a particular player. Here, the system streamed the cylindrical panorama video to the remote clients which extracted a perspective-corrected camera view on the client side. However, popular soccer games often attract millions of concurrent users. For example, during the 2014 FIFA World Cup in Brazil, the web player and app had during the 56 games streamed video to about 24 million unique users [4]. If we additionally take into account the potential number of TV viewers, e.g., 909.6 million television viewers tuned in to at least one minute of the 2010 FIFA World Cup final at home [11], we definitely have a large challenge in providing real-time virtual camera services in such a scale using our previous approach of sending the entire, full-quality panorama to every user.

There are generally two approaches to manage the virtual camera. The first as we have presented earlier where the entire panorama is sent over the network and the virtual view is extracted at the client side. The second alternative performs the generation of the view from the panorama on the server side sending only the virtual view video over the network. Thus, the trade-off on the server-side is between processing and outgoing bandwidth, and vice-verse on the client side.

In general, the de facto streaming approach using segmented, adaptive HTTP streaming has proven to scale well. We have therefore adopted this solution in our system, and in this demonstration, we present a system where the panorama video is divided into tiles, i.e., each encoded in multiple qualities (and thus bit rate). Then, the panorama video tiles are retrieved by the client in a quality depending on the current position and the behaviour of the virtual camera, i.e., the video quality of the tile changes dynamically according to the user interaction, and the panorama is restored on the client side with different qualities in different areas of a frame.

Our initial experiments indicate that there is a large potential for saving bandwidth on the cost of trading quality in areas of the panorama frame not used for the extraction of the virtual view. The proposed demonstration therefore shows how the client performs and how the quality of the extracted view and the panorama video changes when the virtual camera moves to another region of interest.

2. THE COSTS OF VIRTUAL VIEWS

We have earlier presented our approach for generating the high resolution cylindrical panorama videos in real-time [8]. We have also demonstrated that these video can be used to generate individual personalized *virtual views* of the game [5]. When it comes to delivering video to the client, we have explored two possibilities with respect to creating virtual views.

Our initial approach is to transfer the entire panoramic video and generate the virtual views on the client. This gives cheap processing requirements on the server-side at the cost of very high bandwidth requirements. In our example system installed at Alfheim stadium, the average size of each 25-fps *3-second* segment of the the 4096×1680 panorama video is approximately 2.1 MB[1], i.e., the bandwidth requirement for each client becomes about 5.7 Mbps merely for the transfer of the panorama video, and in future systems, a much higher resolution panorama is desirable. Then, after the panorama is successfully transferred, the client needs to process it so that a virtual view can be extracted. Earlier we demonstrated that this can be accomplished in real-time on commodity graphics hardware [5], and figure 2 demonstrates the performance as a function of output resolution. These values are computed for extraction of virtual view on a GPU. Thus, the bandwidth requirement is quite high, but processing wise, the client devices manage the load.

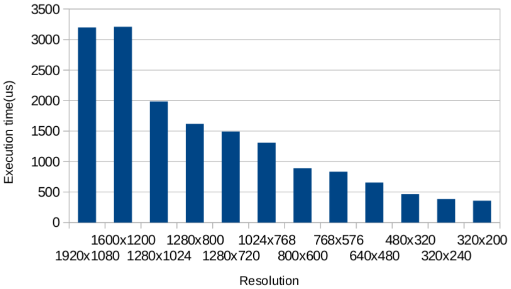

Figure 2: Execution times for various sizes of virtual camera on GTX 460

An alternative approach is to generate the virtual views on the server and only stream the generated virtual view video to the client. Thus, in this approach, the reprojection is performed on the server side. This approach requires nothing more than a browser that is capable to play a video on the client device, i.e., it severely reduces the computational load and the bandwidth requirements on the clients. However, the processing costs on the server-side are huge, and it quickly becomes a large bottleneck as not only must we generate the virtual view, but we must also encode the video for compression. We have made a few experiments using the second generation hardware from Nvidia [14]. Our experiments show that the GeForce GTX 750 Ti GPU can encode 16 full HD video streams at 30 frames per seconds [14]. Experiments showed that this was the limiting factor in how many unique views we could create in real-time. This implies that if we want to provide a service to say 100,000 concurrent users, we would require a cluster totaling to about 6,250 GPU devices. Such an initial installation costs at the time of writing about 937,500 USD merely for the GPUs.

Owing to the challenges mentioned above, no straight forward solution is going to work well for scaling our system to large numbers of concurrent users. However, as the HTTP streaming solutions have proved to scale well from a sending-side point of view using for example CDNs, we have looked at solutions for the first approach – client side generated virtual views.

3. SYSTEM OVERVIEW

Based on the decision in the previous section, the challenge is to reduce the cost of streaming a complete panorama video to every users. In this respect, researchers in the multimedia community have for some time analyzed region-of-interest streaming solutions. For example, tiling is discussed in [3, 2, 7, 12, 13, 16, 18]. Furthermore, [1, 10, 17, 19] extensively address the problem of automatically generating personalized content, and [9] discusses plain cropping. However, our target is a large scale solution scaling the delivery using modern HTTP streaming where each user independently interacts with the system to have a personalized view using zoom, pan and tilt, i.e., the entire panorama must be retrieved and the quality of the tiles are based on the per user interaction.

[1] This number depends on the lighting and weather conditions, but the given number works well as an example.

Similar to many other approaches, our solution is based on dividing the panorama into tiles as shown in figure 3, each encoded as an adaptive HTTP stream using for example HLS. A client retrieves segments in as high quality as possible for segments being used for the virtual camera, and the rest of the tiles are retrieved in decreasing lower quality depending on the distance to the edge of the virtual camera image. In contrast to for example [16] retrieving only tiles in the region of interest, we need to retrieve all tiles since the virtual camera moves and at least low quality data needs to be available if the user zooms out or moves quickly. Another difference is that the tiles fetched do not follow a strict logic apart from being in the neighborhood of the current tile. In [18], for instance, all the tiles are being fetched, but the reduction in quality is reflected by salience. Moreover, the non-linear nature of a panorama-virtual view transformation introduces further complexities in the approach. For example, in figure 3 it can be seen that the collection of required tiles do not form any simple shape like a rectangle or a square, e.g., as used in [9]. This poses different challenges than the ones that are being tackled in for example [12] where the panning and tilting corresponds to strictly navigating the panorama along the horizontal and vertical directions respectively. Such an approach adds complexity on the tile retrieval strategy as the quality adaption strategy not only must take into account available network bandwidth and client device resources (as usually done for *one* stream), but it must also coordinate the tile qualities according to the dynamic position of the virtual camera.

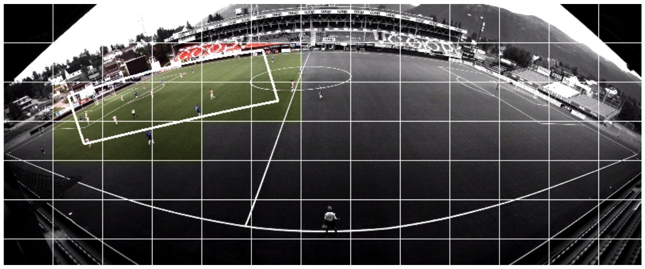

Figure 3: Example of panorama tiling (320x256px)

4. EVALUATION

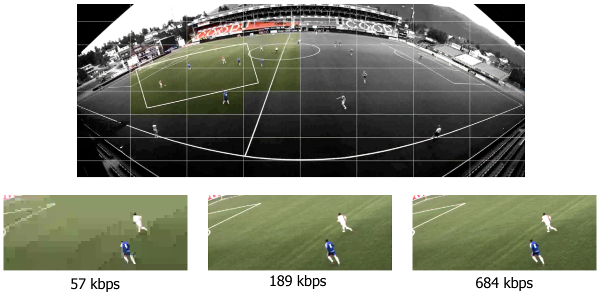

Figure 4: Tiles in different quality

Above, we said that the full quality panorama video required about 5.6 Mbps. If we divide the panorama in 8 × 8 tiles as shown in figure 4 with the given tile qualities (and bitrates), a complete full-quality panorama requires 8.6 Mbps due to the loss of compression across tile boundaries. However, if the user zooms as shown in the figure requiring only full quality for 10 of the tiles (the colored tiles used for the virtual view), the respective bandwidth requirements of the panorama decreases to 3.2 and 2.0 Mbps when using the middle and low quality for rest of the tiles (gray).

Figure 5 shows an example of how our virtual viewer works. It can be observed that the virtual view presents no loss in quality. However, the parts of panorama that are not being shown in the virtual view are fetched in the lowest possible quality. This phenomenon is quite evident in the preview image.

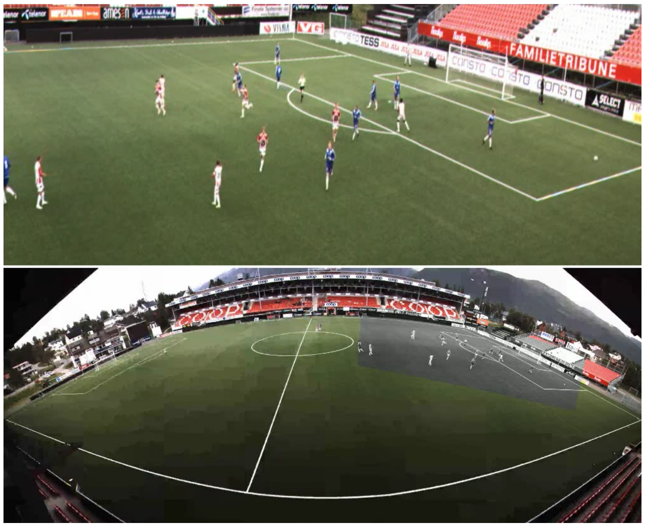

Figure 5: An example of a multi quality tiled-panorama and the virtual view that is extracted from it. The quality of the panorama is quite poor in the areas that are not being shown to the user. However if the user decides to pan quickly, she still gets a reliable low quality video instead of a black patch.

Such a system comes with a rather interesting trade-off with respect to the segment size. A segment of *3 seconds* compared to 3 segments of *1 second* each has a certain advantage in the encoded file-size due to the reduction in the number of I-frames. However, the segment size also determines how quickly a tile can change it's quality. Our initial experiments showed that this trade-off is an interesting one to study. Since the virtual camera moves, it is hard to see the differences of lower quality tiles if the levels are not too far apart. However, the user interaction with in a 3-second period can be assumed completely random and it cannot benefit from the predictive model as much as a 1-second segment could.

5. DEMONSTRATION

In this demo[2], we present a system for real-time interactive zooming and panning of panorama video using video from real-world installations in two Norwegian soccer stadiums. We show how the quality changes of different parts of the panorama video when moving the virtual camera.

[2] http://home.ifi.uio.no/vamsidhg/mmsysDemo

We also show that if there are large differences between the quality layers, reduced quality is noticeable when quickly moving the virtual camera, but if the layers are carefully selected, but still saving bandwidth, it might be hard to see the quality differences due to the view movement. Thus, the tiling approach has potential to greatly reduce the required bandwidth of a scenario where every user is his or her own cameraman [6].

6. REFERENCES

[1] N. Babaguchi, Y. Kawai, and T. Kitahashi. Generation of personalized abstract of sports video. In *Multimedia and Expo, 2001. ICME 2001. IEEE International Conference on*, pages 619–622, Aug 2001.

[2] F. Chen and C. De Vleeschouwer. Personalized production of basketball videos from multi-sensored data under limited display resolution. *Comput. Vis. Image Underst.*, 114(6):667–680, June 2010.

[3] E. Foote, P. Carr, P. Lucey, Y. Sheikh, and I. Matthews. One-man-band: A touch screen interface for producing live multi-camera sports broadcasts. In *Proc. of ACM MM*, pages 163–172, 2013.

[4] Fédération Internationale de Football Association. 2014 FIFA World Cup breaks online streaming records. http://www.fifa.com/aboutfifa/organisation/-news/newsid=2401405/,
2014.

[5] V. R. Gaddam, R. Langseth, S. Ljødal, P. Gurdjos, V. Charvillat, C. Griwodz, and P. Halvorsen. Interactive zoom and panning from live panoramic video. In *Proc. of ACM NOSSDAV*, pages 19:19–19:24, 2014.

[6] V. R. Gaddam, R. Langseth, H. K. Stensland, P. Gurdjos, V. Charvillat, C. Griwodz, D. Johansen, and P. Halvorsen. Be your own cameraman: Real-time support for zooming and panning into stored and live panoramic video. In *Proc. of ACM MMSys*, pages 168–171, 2014.

[7] R. Guntur and W. T. Ooi. On tile assignment for region-of-interest video streaming in a wireless LAN. In *Proceedings of the 22nd international workshop on Network and Operating System Support for Digital Audio and Video - NOSSDAV '12*, page 59, New York, New York, USA, 2012. ACM Press.

[8] P. Halvorsen, S. Sægrov, A. Mortensen, D. K. Kristensen, A. Eichhorn, M. Stenhaug, S. Dahl, H. K. Stensland, V. R. Gaddam, C. Griwodz, and D. Johansen. Bagadus: An integrated system for arena sports analytics – a soccer case study. In *Proc. of ACM MMSys*, pages 48–59, Mar. 2013.

[9] R. Heck, M. Wallick, and M. Gleicher. Virtual videography. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 3(1):4–es, Feb. 2007.

[10] R. Kaiser, M. Thaler, A. Kriechbaum, H. Fassold, W. Bailer, and J. Rosner. Real-time person tracking in high-resolution panoramic video for automated broadcast production. In *Proc. of CVMP*, pages 21–29, 2011.

[11] KantarSport. 2010 FIFA World Cup South Africa - Television Audience Report. http://www.fifa.com/mm/document/affederation/tv/-01/47/32/73/2010fifaworldcupsouthafrica-tvaudiencereport.pdf,
2010.

[12] A. Mavlankar and B. Girod. Video streaming with interactive pan/tilt/zoom. In M. Mrak, M. Grgic, and M. Kunt, editors, *High-Quality Visual Experience*, Signals and Communication Technology, pages 431–455. 2010.

[13] K. Q. M. Ngo, R. Guntur, and W. T. Ooi. Adaptive encoding of zoomable video streams based on user access pattern. In *Proceedings of the second annual ACM conference on Multimedia systems - MMSys '11*, page 211, New York, New York, USA, 2011. ACM Press.

[14] NVIDIA. NVIDIA - NVIDIA hardware video encoder. `http://developer.download.nvidia.com/compute/nvenc/v4.0/NVENC_AppNote.pdf`, 2014.

[15] S. Sægrov, A. Eichhorn, J. Emerslund, H. K. Stensland, C. Griwodz, D. Johansen, and P. Halvorsen. Bagadus: An integrated system for soccer analysis (demo). In *Proc. of ICDSC*, Oct. 2012.

[16] A. Shafiei, Q. M. K. Ngo, R. Guntur, M. K. Saini, C. Pang, and W. T. Ooi. Jiku live. In *Proceedings of the 20th ACM international conference on Multimedia - MM '12*, page 1265, New York, New York, USA, 2012. ACM Press.

[17] X. Sun, J. Foote, D. Kimber, and B. Manjunath. Region of interest extraction and virtual camera control based on panoramic video capturing. *IEEE Transactions on Multimedia*, 7(5):981–990, 2005.

[18] H. Wang, V.-T. Nguyen, W. T. Ooi, and M. C. Chan. Mixing tile resolutions in tiled video: A perceptual quality assessment. In *Proceedings of Network and Operating System Support on Digital Audio and Video Workshop*, NOSSDAV '14, pages 25:25–25:30, New York, NY, USA, 2013. ACM.

[19] R. Xu, J. Jin, and J. Allen. Framework for script based virtual directing and multimedia authoring in live video streaming. In *Multimedia Modelling Conference, 2005. MMM 2005. Proceedings of the 11th International*, pages 427–432, Jan 2005.