Streaming Location-based Panorama Videos into Augmented Virtual Environment

Yi Zhou, Peifu Liu, Jingdi You, Zhong Zhou
State Key Laboratory of Virtual Reality Technology and Systems
Beihang University
Beijing, China
zhouyi@vrlab.buaa.edu.cn

Abstract

Location-based panorama systems such as Google Street View let users explore places around the world through panoramic bubbles or strips. The panorama image is easy to be deployed, but it can only provide the static views of capturing time and lacks developing process. In this paper, we present an augmented virtual environment system that combines multiple location-based panorama videos with the structural context of scenes. The raw panorama images are from several independent video cameras. A frame synchronization method of video streams is proposed to provide the temporal consistency in the panorama stitching. Our novel method augments the virtual environment through mixing it with the panorama videos. To the best of our knowledge, this is the first paper to fuse panorama videos with virtual environments. The system is demonstrated in a campus-wide area, and it enhances users' walk-through experiences in the experiment environment.

Keywords: Panorama video, augmented virtual environment, Video streaming synchronization

1. Introduction

The ability to visualize the real world in a virtual environment is long-cherished. 360-degree panoramas [1] provide an immersive experience for observers through simultaneous camera shots and reliable automated stitching. Nowadays, location-based panorama systems are prevailing and enable users to explore places around the world through panoramic bubbles or stripes. Google Street View [2] produces a street view imagery with discrete panoramic bubbles. And Microsoft Street Slide [3] employs multi-perspective strip panoramas to generate a visual summary of continuous street sides.

Unfortunately, the image source used in these systems is a sequence of offline-stitched pictures that only show users a history imagery. It cannot reflect the scene variations. Moreover, existing systems lack overall sense from other view than the given points and Street Slide provides only a flat summary of two sides.

The concept of combining the location-based videos with virtual environment attracts a lot of attention. Surveillance systems use visualization techniques, such as texture projection, to provide a global context which helps users understand the spatial relationships between multiple videos. The typical technique, video texture projection, suffers from severe foreshortening and zigzag while user left the original viewpoint. But it enhances users' visual experiences in a particular view range.

In this paper, we present an augmented virtual environment system that combines multiple location-based panorama videos with the virtual environment. The system extends panoramic bubble's immersive nature and also gives an overview sensation for discrete videos. Our novel method augments virtual environment through mixing with the panorama videos. The dynamic panorama videos are visualized by depth order.

We also present a video streaming synchronization method to provide the temporal consistency in the panorama stitching. Our stitching procedure is accomplished by blending the textures of aligned bubble meshes during the render loop.

To the best of our knowledge, this paper is the first trial to combine panorama videos with virtual environment. And it includes three contributions to panorama streaming and rendering. The elements of our system include:

- A user-controlled streaming synchronization method that provides the temporal consistency in the panorama stitching,
- A novel mixed visualization method that provides the spatial consistency, and
- The campus-level 3D surveillance framework with workload control.

2. Related work

Augmented virtual environment (AVE) [4] is proposed to integrate multiple videos into a 3D context model. With the help of 3D models, street lines, site icons and other

annotations, it is intuitive and convenient for users to understand objects and activities in the multiple videos. And a user-friendly graphics interface that supports navigating or dragging will provides a good immersive in the virtual environment. Moezzi et al.'s seminal work [5] on multi-perspective interactive video (MPI-Video), extracted object from multiple videos at the same location and used it to augment the virtual environment.

Sawhney et al. [6] further developed the thought of integrating by projecting multiple videos onto a 3D environment model. They thought that, if the video is projected from the actual camera location with the correct camera parameters, the walls and floors in the video can seamlessly match the model. Video Flashlight system [6] demonstrated texture projection's feasibility. They detected moving objects inside the video and visualized them as textured dynamic rectangles moving around in the 3D model. Several other similar motivated surveillance applications have also been introduced [7] [8].

Methods for large-scale environment, like city-level, are presented along with the growth of videos. They employed geographical approaches to simulate or augment virtual objects in 3D environments. Kim et al [9] introduced different approaches to analyze the videos of cities under differing conditions and then created augmented Aerial Earth Maps (AEMs) with live and dynamic information. Austin et al [10] implemented the registration of 3D Map and hundreds of web cameras by using corresponding points.

Other devices like PTZ or fish-eye cameras are also used for 3D integrating, such as [10] [11]. Our work is based on integrating multiple panorama videos with the structural context and there are few previous works focusing on this topic.

The classical synchronization of multiple streaming is mainly about the audio-video synchronizing or NTP-based rough matching of video streams, which is widely used in video conference, video surveillance and video live broadcast systems [12]. Since current IP cameras don't provide time synchronization services, existing AVE systems merge the pictures from different time into one scene. It will affect the scene completeness in rendering esp. in the same bubble when there are several streams inside the panorama video.

Shrestha et al. [13] presented a synchronization method based on detecting flashes which are presented in the video content. The flashed frames are selected as benchmark and easily detected by using an adaptive threshold on luminance variation across the frames. They [14] also describe an audio-fingerprints matching method to synchronize videos with complete audios. They both are used for adjacent videos synchronization. Yan et al. [15] relies on correlating space-time interest point distribution in time between videos. Space-time interest points represent events in videos that have high variation in both space and time.

Our synchronization method is based on user- identified event, such as a flash event. The synchronization baseline is streamed to the client as a timestamp. Each independent video is played according to the synchronized buffer queues.

3. System architecture

Our open extensible system is comprised of three main components, and is depicted in Fig. 1: (1) the client, as the consumer of our system, can be PC application, web browser, or mobile application. The client obtains services and data, such as video streams from the streaming server; (2) several servers, the service provider, consist of GIS server, web server, and several streaming servers. The GIS server and streaming server provide data for visualization, while web server offers the fundamental data. The streaming data is synced by user-identified event (details in section 4); and (3) underlying data source is composed of web cameras, playback files, and the database. The important components are described in the following sections.

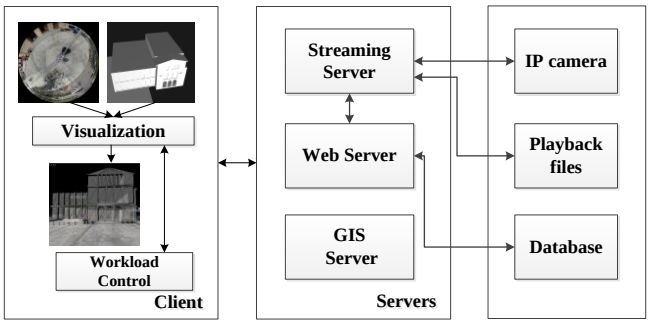
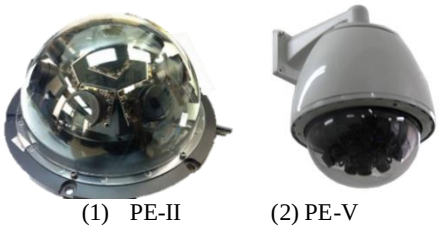


Fig. 1. System architecture

IP Camera: The panorama video capturing uses the Panoeye series panoramic surveillance cameras [16], shown in Fig. 2. The high resolution PE-V hemispherical digital video camera system has eight 1.3 MP cameras that enable the system to collect video from a hemispherical view with 360 degrees in horizontal and 180 degrees in vertical, and a standard Ethernet interface with locking screw connection that allows almost 10MP resolution videos to be streamed to disk at more than 15fps. A PE camera can output 8*2 real-time video streams encoded in H.264. For the PE-II camera, each lens provides a D1 main stream and a CIF auxiliary stream.



(1) PE-II (2) PE-V

Fig. 2. PE panoramic monitoring camera

GIS Server: GIS is one of the effective tools to manage large-scale scene data. The location of our outdoor cameras is expressed as exclusive latitude-longitude. Based on this

location, it is easy to find more information of the real world for AVE, including 3D models and annotations, which greatly extends our system's application range. Our system uses the 2D maps from the GIS server to locate the panorama camera and the 3D models to augment the panorama video.

Web Server: As the director of the whole system, web server ensures that the whole system runs exactly as the administrator expects. System administrator can handle the system through the administration interface. On the other hand, it can provide kinds of services, such as delivering HTML file to web browser or XML file to windows application, through the HTTP protocol to help the admin understand the running state of the system. Then the client can decide whom to send a request to. Furthermore, our web server provides authority authentication service to satisfy the privacy requirement in surveillance systems.

4. Multi-video acquisition & synchronization

Our video streams are captured by physical cameras or generated from playback files. The video streams are played in one virtual environment and need to be synchronized. The section 4 introduces the architecture of the streaming server and its synchronization method, showed in Fig. 3 and Fig.4 respectively.

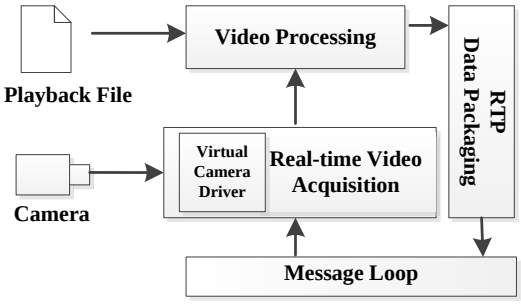


Fig. 3. Streaming server architecture

As described in Fig.3, the streaming server uses a message loop to handle RTSP request. If the client requests live videos, the streaming server controls the real-time video acquisition getting data from physical cameras via the virtual camera driver and sending for next processing. The virtual camera driver encapsulates various cameras into a virtual camera and provides a unified programmable interface. In this way, the streaming server is able to receive data flows from different kinds of cameras without concerning specific camera drivers. The new types of physical camera can be easily added to the system. If the client requests playback, the video processing module is called immediately to load record files. This module abstracts the process procedure, which is independent from their detailed coding formats. After processed, frames are packed into RTP packages and flowed to the message loop for delivering.

Although cameras usually provide respective timestamps, they hold no common time benchmark. Furthermore, the frame rate of a given camera may not be the same as others'. So it's almost infeasible to synchronize simply by using timestamp. We provide a synchronization method consisting of server part and client part, introduced in Fig. 4. The server recalculates timestamp, while the client calculates serial numbers and uses them to synchronize the streams.

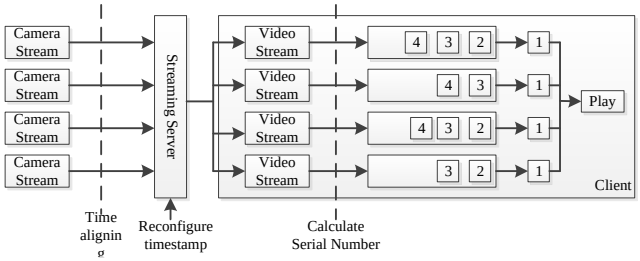


Fig. 4. Time aligning and synchronized playing.

Given the camera set $C=\{C_k \mid K=0,1,2,\cdots\}$, the server offers a tool for manually aligning their time benchmark(see Fig. 5). A camera C_0 is chosen and its time is used as the benchmark. Then the operator chooses another camera C_k and finds when an event occurs in both cameras. In Fig. 5, the operator notices a flash in several camera views. Then the user tries to find out the moment the flash first appeared by manual adjusting in each view. After aligning, the time offset T_d can be calculated through their timestamps

When a frame of C_k arrives at the server, its timestamp is recalculated before delivered by using T_d. Then these frames arrive at the client and are synchronized by using following steps:

Step1: Denote reference frame rate F=min F_k and create a timeline T whose origin point is T_0, and $T_0<=TS$, where F_k is the fps of C_k and TS is the timestamp of the first arrived frame.

Step2: Given a frame f_m of C_k and its timestamp TS_{mk}, its serial number is calculated as:

$$SN(f_m, C_k) = \lfloor F * (TS_{mk} - T_0) \rfloor$$

Then the frame is sent to corresponding receiving buffer for camera C_k.

Step3: The playing component starts a timer and gets the next frames. Each camera holds a state indicating whether it is obstructed or not, and if the camera is obstructed, the playing component will ignore its frames.

We extract the minimum serial number of the header frames from unobstructed cameras. Then take respective frames from receiving buffers for synchronous playing. However, if a receiving buffer is empty, a timer is started to wait for the desired frame. If the corresponding frame arrived before timeout, it is sent to play together with other frames, or else it will be ignored at this time. If a camera is always in timeout for a certain period, it will be set obstructed.

Fig. 5: Interface of aligning tool. A flash occurs in several cameras.

5. Mixing panorama videos with 3D virtual environments

The mixing method contains the registration of panorama videos and the environment, a visualization to combine all manners of models and video in a coherent visualization to improve the understanding, and a workload control strategy. Our registration method, which is based on Perspective-n-Point theory and similar to Austin's [10], is employed in our system. The details about registration are not presented here because of space limit. More details of the latter two topics are introduced below.

5.1. Visualization

We define three views to satisfy all of the different situations. Fig. 6 show the view switch graph and its effects.

Fig. 6. View switch: the outside view summarizes the bubbles, while the inside view and across view are close-up views.

5.1.1. Inside bubble view: Static view

Our system achieves video overlaying on reference models by dynamically deciding the render tree. We define a render tree, which has a dramatically influence on results, to decide the render order of bubbles and other models. As Fig. 7 shows, the leaf nodes are traversed based on its hierarchy. But the bubbles are set to render in an order based on its depth under the current viewport. The bubbles and models use alpha masks to realize a blend or transparency effect.

Inside a bubble, the usual viewpoint is the optic center of panorama. If the user want to leave this view, it must long press the direction button until the scaling factor exceeds the threshold, otherwise the viewpoint will stay at the center of bubble.

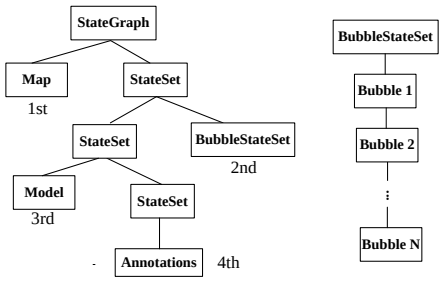


Fig. 7. The render tree of blend effect

5.1.2. Across bubble view: View transition

View transition between two discrete bubbles has great influence on immersive of the whole virtual environment. We suppose that this case only happens between panorama bubbles with overlapping views and motions, otherwise we use a view leap method to instead of it.

Unlike the technique used in systems such as Google Street View, we use the image fusion between the two bubbles by projecting panorama texture to the complete scene. The texture projection method of the panorama is based on the shadow mapping technique of a zero-decrement point light source. The render loop contains a pre-render pass and a nested-render pass. In the pre-render pass, a 24-bits depth map is generated by spherical projection. And it will be used for depth test in the second pass. These points who are not occluded will be textured with panorama pixels.

We search for the target bubble and change the bubble view into projection mode. During this mode, the view point changes along the line between centers of these bubbles. We refer to this view as virtual view and the angle between the pixels in view line and its virtual view ray as θ. And two weight are computed to decide the blend effect, as Fig.8 showed.

$$\omega_2 = \theta_1/(\theta_1 + \theta_2), \omega_2 = \theta_1/(\theta_1 + \theta_2)$$
$$I_p = \omega_1 * p_1 + \omega_2 * p_2$$

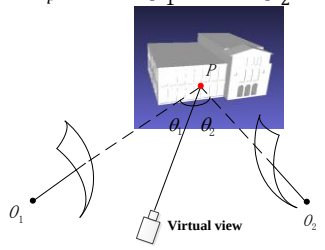


Fig. 8. Image blend based on angle

5.1.3. Outside bubble view: Visual summary

In this view, all videos are showed as a 3D hemisphere. Its location and rotation are generated by our calibration tool, while the scale is set by experience. There is no more limit to the eye location, and users can observe multiple individual bubbles from an arbitrary viewpoint. In this case, a bubble is more like a point which can be selected and stepped into.

5.2. Workload control

While browsing 3D environment, some bubbles are occluded or small enough on the screen, and we do not need to update its texture or use lower resolution texture. Our streaming server supports double resolutions of videos and we implement dynamic resolution transition and workload control which refers to the bandwidth.

We also employ a view-dependent method to toggle between the main stream and the auxiliary stream. As show in Fig. 9, the selection depends on the bubble center's distance *d* to the viewpoint. Then, we use the result of occlusion query to decide whether to render the hemisphere.

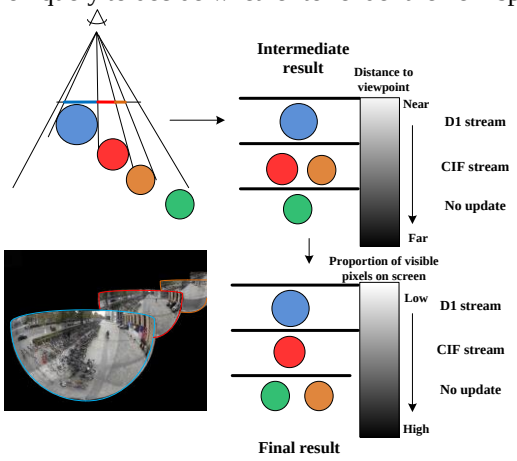


Fig.9. The view-dependent client workload control

As we mentioned above, the panorama videos are rendered on a hemisphere mesh. According to the result of occlusion query, these bubbles which are not in the view frustum should be removed from streaming list. And these bubbles whose visible pixels are lower than the threshold should disconnect from the server or toggle the auxiliary video stream.

6. Experiment evaluation

We carried out experiments to evaluate out our streaming server's performance and two proposed methods. All the tests are conducted on a PC workstation with a NVIDIA Quadro 2000 graphics card, double Intel(R) Xeon(R) X5680 at 3.33GHz, 20GB memory, and a 100Mbps Ethernet connection to the campus network. In addition, five outdoor PE-V cameras, one indoor PE-II camera and several virtual cameras are used in our system.

6.1. Streaming server performance and synchronization

The average response latency in different load conditions is tested and showed in Fig. 10. The result shows that the number of the stream links ranges from 32 to 320, while the response latency ranges from 6.63ms to 10.12ms. In the concurrency test, we see a rapid reduction when the resolution increased dually or more (higher than CIF).

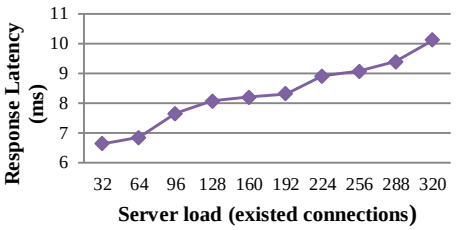


Fig. 10. 10*10 D1 video tests on the different load conditions and the respective average latency.

Table 1. Concurrency performance

Exp. term	1	2	3	4	5
Resolution	QCIF	CIF	640*480	D1	1080P
ABR(kbps)	34.41	132.17	768.75	912.43	2845.29
Max Links	618	601	372	354	118

Our synchronization method is demonstrated by the two lenses from one camera. In Fig. 11, a ghosting stitching occurs when people walking across their overlaying region (Fig.10 (b)). With our method, motion lags are significantly reduced between cameras.

(a) (b) (c)

Fig.11. synchronization results: (a) overlaying region (b) ghosting occurs (c) synchronous motion

6.2. Visualization and workload control

Our client is implemented in C++ using OSG and GLSL shader. The videos are received over the network in separate threads. The decoding threads take up the major system resources. The visualization result is showed in Fig.13.

We simulate an animation path of virtual environment to evaluate the workload control. Except for five outdoor cameras, three playback files are used in our test. Fig. 12 shows the bandwidth occupation under different resolutions.

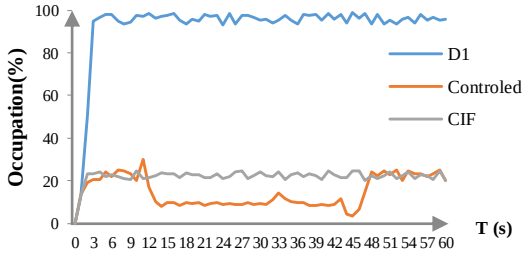


Fig. 12. Bandwidth occupation under different resolutions. Each D1 video occupies a bandwidth of about 1MBps and CIF 250KBps.

In Fig. 12, the occupation of the three situations soon soars to a summit when getting close to the ground. With the view-

dependent control, the average occupation is much lower than CIF, thus more cameras can be deployed in our system simultaneously.

7. Conclusion and future work

In this paper, we introduced methods for Augmented Virtual Environment (AVE) with multiple panorama videos. The proposed visualization solution creates a global sense of multiple bubbles, and users can observe them at any view. The system presents a frame synchronization to provide temporal consistency in the panorama stitching. The result makes it clear that our work provides a more engaging virtual environment at least for the given test.

It is also important to note a few of our limitations. First, the method of view transition does not consider dynamic tone mapping between two cameras. Secondly, our solution does not support the automatic camera pose correction which is caused by some unavoidable nature force.

In our future work, we aim to overcome the above limitations, and engage on the fusion method of ordinary videos and panoramic videos.

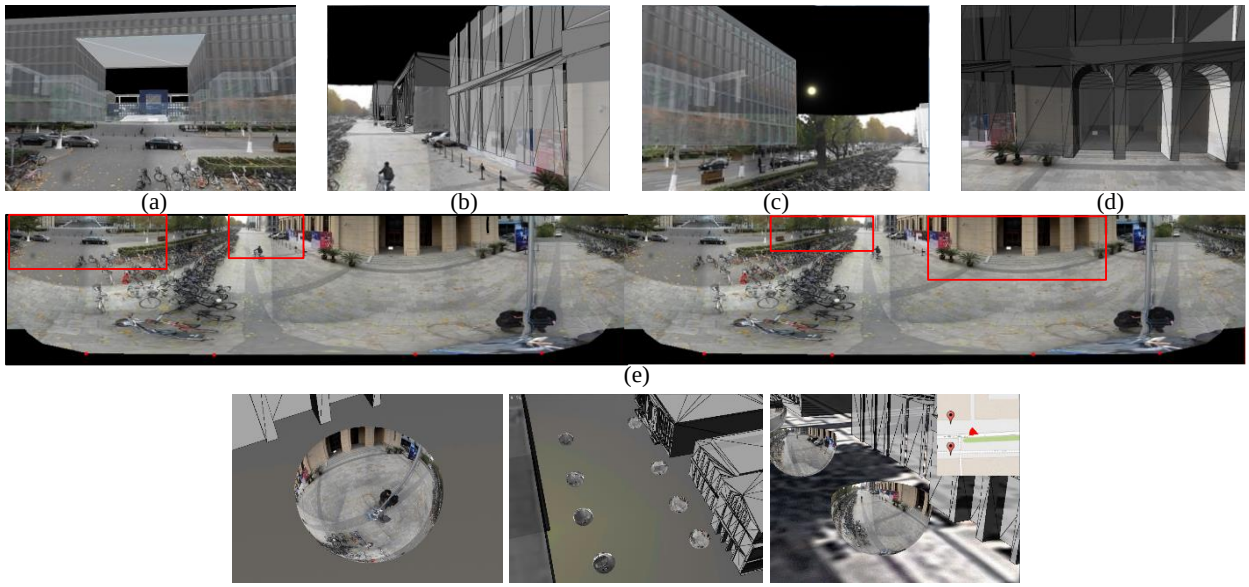


Fig.13. Results from our system using 8 PE-V 200 camera (three of them are virtual cameras), 64 D1 steams, 64 CIF streams: (1) inside view: (a) ~(d) view around the bubble, (e) the stitched spherical panorama (2) outside view: (f) single camera; (g) eight cameras along the road; (h) an assistant minimap for location

References

[1] S. E. Chen, "Quicktime VR: An image-based approach to virtual environment navigation," Proc. *the 22th ACM Annual Conference on Computer Graphics and Interactive Techniques*, Los Angeles, pp. 29-38, August 1995.

[2] L. Vincent, "Taking online maps down to street level," *Computer*, vol. 40, no. 12, pp. 118-120, December 2007.

[3] J. Kopf, B. Chen, R. Szeliski, et al., "Street slide: browsing street level imagery," *ACM Transactions on Graphics*, vol. 29, no. 4, pp. 96, July 2010.

[4] I. O. Sebe, J. Hu, S. You, et al., "3D Video Surveillance with Augmented Virtual Environments," Proc. *the 1st ACM SIGMM International Workshop on Video Surveillance*, California, pp. 107-112, November 2003.

[5] A. Katkere, S. Moezzi, D. Kuramura, et al., "Towards video-based immersive environments," *Multimedia Systems*, vol. 5, no. 2, pp. 69-85, March 1997.

[6] H. S. Sawhney, A. Arpa, R. Kumar, et al., "Video Flashlights: Real Time Rendering of Multiple Videos for Immersive Model Visualization," Proc. *the 13th Eurographics Workshop on Rendering*, Pisa, pp. 157-168, June 2002.

[7] P. M. Roth, V. Settgast, P. Widhalm, et al., "Next-generation 3D visualization for visual surveillance," Proc. *the 8th IEEE International Conference on Advanced Video and Signal-Based Surveillance*, Klagenfurt, pp. 343-348, August 2011.

[8] G. H. de, J. Scheuer, R. V. de, et al., "Egocentric navigation for video surveillance in 3d virtual environments," Proc. *IEEE Symposium on 3D User interfaces*, Lafayette, pp. 103-110, March 2009.

[9] K. Kim, S. Oh, J. Lee, et al., "Augmenting aerial earth maps with dynamic information," Proc. *the 8th IEEE*

International Symposium on Mixed and Augmented Reality, Orlando, pp. 35-38, October 2009.

[10] A. D. Abrams, R. B. Pless, "Webcams in context: web interfaces to create live 3D environments," Proc. *the International Conference on Multimedia*, Firenze, pp. 331-340, October 2010.

[11] P. DeCamp, G. Shaw, R. Kubat, et al., "An immersive system for browsing and visualizing surveillance video," Proc. *the International Conference on Multimedia*, Firenze, pp. 371-380, October 2010.

[12] A. P. Tresadem, I. D. Reid, "Video synchronization from human motion using rank constraints," *Computer Vision and Image Understanding*, vol. 113, no. 8, pp. 891-906, August 2009.

[13] P. Shrestha, H. Weda, M. Barbieri, et al., "Synchronization of multiple video recordings based on still camera flashes," Proc. *the 14th Annual ACM International Conference on Multimedia*, Santa Barbara, pp. 137-140, October 2006.

[14] P. Shrestha, M. Barbieri, H. Weda, "Synchronization of multi-camera video recordings based on audio," Proc. *the 15th International Conference on Multimedia*, Augsburg, pp. 545-548, September 2007.

[15] J. Yan, M. Pollefeys, "Video synchronization via space-time interest point distribution," Proc. *Advanced Concepts for Intelligent Vision Systems*, Brussels, pp. 501-504, August 2004.

[16] http://www.panoeye.com.cn/