Video Surveillance using Augmented Virtual Environments

Aleksandar Lj. Milosavljević[1], Aleksandar M. Dimitrijević[2] and Dejan D. Rančić[3]

Abstract – **In this paper we present video surveillance technique based on Augmented Virtual Environments (AVE). AVE represents 3D virtual environment augmented with multiple video streams that are fused with 3D models in a real–time. This approach enables an observer to comprehend real–world video from arbitrary views of the scene. Our solution is based on use of a 3D GIS as the virtual environment. To enable registration of video frames into the 3D GIS we proposed a method for camera view modeling applicable to PTZ video cameras.**

Keywords – **Augmented Virtual Environments, Video Surveillance, Geographic Information Systems, Virtual Reality.**

I. Introduction

The ability of GIS to handle and process both location and attribute data distinguishes it from other information systems. It also establishes GIS as a technology that is important for a wide variety of applications [1]. Traditionally, the majority of geographic information systems were limited to the visualization of geospatial data in two dimensions (2D GIS). The fact that we relate to our world in three or more dimensions suggests that some types of data may be more readily visualized and analyzed in 3D [2]. With the development of graphics hardware, virtual reality techniques originally developed for interactive computer games are exerting more and more influence in the field of 3D GIS. Li *et al.* [3] expressed the need for 3D GIS for urban environments in order to understand the 3D landscape with many high buildings.

Research presented in this paper deals with the integration of 3D GIS and video surveillance systems. Real–time video monitoring is playing an increasingly significant role in surveillance systems in various security, law enforcement, and military applications [4]. A typical outdoor urban surveillance system consists of multiple cameras overlooking different areas. However, conventional video monitoring systems have various problems with multi–point surveillance [5]. A typical system of conventional video monitoring connects each video camera directly to a corresponding display screen. Therefore, we have as many screens as video cameras. In these kinds of systems, serious problems can occur when the scale of the monitoring system grows larger than human capacity. Security personnel must mentally map each surveillance monitor image to the corresponding place in the real world, and this complicated skill requires experience and training [5]. To enable multi–camera coordination and tracking, Sankaranarayanan and Davis [6] emphasized the importance of establishing a common reference frame to which each of these cameras can be mapped. They suggested the use of GIS as a common frame of reference because it not only provides a solid ground truth, but more importantly provides semantic information (e.g., locations of roads, buildings, sensitive areas, etc.) for use in applications such as tracking and activity analysis.

Addressing the problem of the human ability (or lack thereof) to successfully fuse and comprehend the information that multi–point video surveillance can provide, Neumann *et al.* [7] proposed a visualization approach based on an Augmented Virtual Environment (AVE). The AVE is a virtual reality model augmented by multiple video streams in real–time to help observers comprehend temporal data and imagery from arbitrary views of a scene.

Our implementation of AVE relies on use of augmented and virtual reality techniques applied to GIS. Augmented reality (AR) aim to combine the real scene viewed by a user and a virtual scene generated by a computer that augments the scene with additional information. Unlike virtual reality (VR), which provides the user with a synthetic environment as a replacement for reality, augmented reality ensures that the user sees the real environment augmented with objects and information from the virtual environment. In order to better understand term “augmented reality”, reality–virtuality continuum defined by Milgram and Kishino [8] should be considered (see Fig. 1). The “real world” and a “totally virtual environment” are at the two ends of this continuum while the middle region is called mixed reality (MR). Augmented reality is near to the real environment side, while augmented virtuality (AV) is closer to the virtual environment side. Unlike augmented reality, augmented virtuality adds real images to virtual environment increasing virtual object's reality degree.

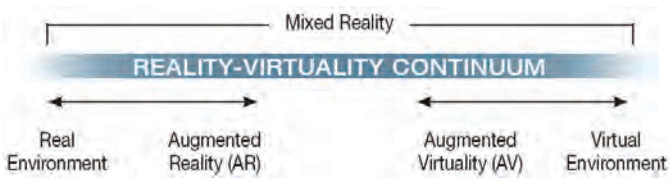

Fig. 1. Reality–Virtuality Continuum [8]

[1]Aleksandar Lj. Milosavljević is with the Faculty of Electronic Engineering, Aleksandra Medvedeva 14, 18000 Niš, Serbia, E–mail: aleksandar.milosavljevic@elfak.ni.ac.rs

[2]Aleksandar M. Dimitrijević is with the Faculty of Electronic Engineering, Aleksandra Medvedeva 14, 18000 Niš, Serbia, E–mail: aleksandar.dimitrijevic@elfak.ni.ac.rs

[3]Dejan D. Rančić is with the Faculty of Electronic Engineering, Aleksandra Medvedeva 14, 18000 Niš, Serbia, E–mail: dejan.rancic@elfak.ni.ac.rs

Both augmented reality and augmented virtuality requires 3D registration of real–world images (in our case video) within virtual environment. Our approach relies on use of 3D GIS as such environment. Conceptual diagram that illustrates integration of GIS and video surveillance is show in Fig 2.

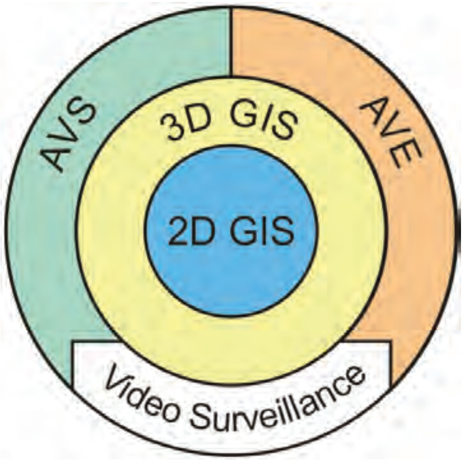

Fig. 2. Conceptual diagram of GIS and Video Surveillance integration

In the proposed conceptual diagram 2D GIS represents a core. It is used for geospatial data management and map visualization. A 3D GIS is built around 2D GIS, i.e. it uses 2D GIS services to enable 3D visualization of geospatial data. Finally, the last shell in this diagram corresponds to systems that integrate 3D GIS with video surveillance. These systems can be divided into two main categories:

- GIS Augmented Video Surveillance (AVS)
- GIS based Augmented Virtual Environment (AVE)

The first category use augmented reality techniques for the integration, while the second is nearer to augmented virtuality. Our previous work [9] included implementation of an AVS system, while in this paper deals with implementation of our AVE prototype *GeoScopeAVE*.

The paper is organized as follows: Section 2 presents a method for registration of camera video into 3D GIS. In section 3 we give an overview of the key features of the implemented prototype. Finally, section 4 presents the conclusions.

II. 3D GIS REGISTRATION OF CAMERA VIDEO

In this section a method for registration of PTZ camera video frames into 3D GIS is presented. PTZ is an abbreviation for Pan–Tilt–Zoom, and in the terminology of video surveillance, it indicates cameras that can rotate in the horizontal (pan) and vertical planes (tilt) and change their level of magnification (zoom). The registration is done in two steps:

- Establishing camera view absolute parameters
- Constructing camera visibility surface

Establishing camera's absolute 3D GIS view parameters require definition of an appropriate view model and transformations from camera's relative coordinate space.

A. Modeling PTZ camera view in 3D GIS

An observer view into the 3D GIS is fully determined by the following seven parameters that can be divided in two groups:

- Position parameters:
  (1) Latitude
  (2) Longitude
  (3) Altitude
- Orientation parameters:
  (4) Azimuth (or yaw)
  (5) Pitch
  (6) Roll
  (7) Field of view (FOV)

When a PTZ camera is in the role of observer, the first group of parameters is fixed and determined by the camera mounting position. Knowing characteristics of the camera lens, the remaining 4 parameters are calculated using the retrieved *pan*, *tilt*, and *zoom* parameters.

The current azimuth, pitch, and roll are determined from the camera *pan* and *tilt* using three calibration parameters:

- Azimuth of the zero–pan camera position ($azimuth_0$),
- Pitch of the north–pan camera position ($pitch_0$), and
- Roll of the north–pan camera position ($roll_0$).

The calibration parameters are determined by the camera mounting. The first parameter ($azimuth_0$) represents the direction of the camera view when the *pan* parameter is set to 0. The other two parameters model small angular deviations in the horizontal plane that arrive from imperfect mounting, and they are represented as pitch and roll when camera is oriented to the north.

After the calibration, all parameters for setting virtual 3D GIS camera orientation are known, and the rotation matrix for aiming the virtual camera is calculated with the following formula:

$$R = R_z(-azimuth_0) \times R_x(pitch_0) \times R_y(-roll_0) \times R_z(-pan) \times R_x(tilt) \tag{1}$$

In the previous formula, R is used to represent rotation matrices. The subscript determines around which axis the rotation is taken, while the parameters in the parentheses are the angles of rotation measured in a counter clockwise direction. Parameters $azimuth_0$, $roll_0$ and $pitch_0$ are previously described calibration parameters, while input parameters *pan* and *tilt* determine the current orientation of the camera in its local coordinate system.

Based on the calculated rotation matrix R, vectors that determine camera orientation in the absolute coordinates of the 3D GIS are calculated with the following formulae:

$$\begin{aligned} \overrightarrow{V_{look}} &= [x_{look} \quad y_{look} \quad z_{look}]^T = R \times [0 \quad 1 \quad 0]^T \\ \overrightarrow{V_{side}} &= [x_{side} \quad y_{side} \quad z_{side}]^T = R \times [1 \quad 0 \quad 0]^T \\ \overrightarrow{V_{up}} &= [x_{up} \quad y_{up} \quad z_{up}]^T = R \times [0 \quad 0 \quad 1]^T \end{aligned} \tag{2}$$

As the subscripts suggest, the first vector V_{look} determines view direction, while the second V_{side} and the third V_{up} determine relative right side and up directions, respectively.

Absolute orientation parameters can be fully determined by two of these three vectors. Formulae that calculate *azimuth*, *pitch*, and *roll* based on V_{look} and V_{side} vectors are the following:

$$azimuth = \arctan\left(\frac{x_{look}}{y_{look}}\right),$$
$$pitch = \arcsin\left(\frac{z_{look}}{\left|\overrightarrow{V_{look}}\right|}\right), \quad roll = -\arcsin\left(\frac{z_{side}}{\left|\overrightarrow{V_{side}}\right|}\right) \tag{3}$$

In order to view a 3D scene in the same way that actual camera do, we had to model the virtual camera with the parameters of sensors used in a real camera. To calculate the angular field of view (*FOV*) based on the sensor's dimension (*S*), a lens's focal length (FL_{min}), and the current *zoom* factor, we used the following formula [10]:

$$FOV = 2 \cdot \arctan\left(\frac{S}{2 \cdot FL_{\min} \cdot zoom}\right) \tag{4}$$

B. Constructing camera visibility surface

Display of camera video in the 3D GIS scene is based on OpenGL texture projection technique [11]. Each video frame updates a texture that is projected into the scene from the camera viewpoint (using `GL_EYE_LINEAR` texture coordinate generation). A common issue of this technique is a dual projection – one along the projector's view direction, and another in the opposite direction. Another issue is a texture being projected onto all surfaces along the projector's view direction, while some of them are not truly visible for the camera viewpoint. To avoid these problems we have developed a method for constructing camera visibility surface to which frame texture projection is only applied. The method is applicable to an arbitrary complex scene. The only limitation is computation intensity which can reduce frame rate for scene rendering.

To construct camera visibility surface the following steps should be applied:

1. Setup the scene view to the absolute camera view
2. Setup the drawing viewport to the camera image size
3. Render the scene with light and textures disabled
4. Read the depth of each pixel in the resulting frame
5. Unproject each pixel with depth value using inverse transformation to determine it's absolute (x, y, z) coordinates
6. Using this 3D matrix create triangles from adjacent points.
7. Calculate each triangle normal and test if the angle between this vector and vector that points to the camera is less that (90º - ε)
8. Create a display list with triangles that passed the test

Steps 1 and 2 are used to setup view to the scene in the exact same way the camera "sees" a real–world. Steps 3 and 4 are needed to acquire depth of the each pixel "seen" by the camera. The scene rendering in the step 3 is optimized for speed and not actually displayed on a screen. Steps 5 and 6 are used to create 3D matrix with absolute coordinates of a surface seen by the camera. To do so we need to unproject all pixels in the frame along with depth information using `gluUnProject` function. Finally, steps 7 and 8 are used to create display list with the camera visibility surface. This display list contains triangles constructed from previous 3D matrix that passed the surface orientaion test. This test is used to eliminate false surface triangles that "connect" different object's truly visible surfaces. The orientation test is illustrated in Fig. 3.

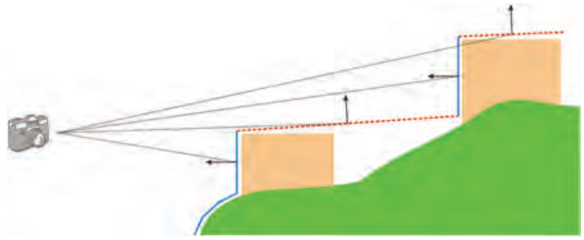

Fig. 3. Elimination of false visible surfaces using the orientation test

The proposed method is compute and data–intensive. In our testing, creation of a visibility surface for 768x576 frame image takes about 0.5–1 second. The resulting display list can contain up to 882050 triangles. Nevertheless, the advantage is that the surface constructed in this way remains valid until the camera orientation or the scene details change.

III. *GEOSCOPEAVE* OVERVIEW

In this section an overview of the developed *GeoScopeAVE* prototype is presented. The application is implemented in MS Visual Studio 2008 as C++ MFC project using OpenGL. For geospatial data access and 2D visualization the application relies on our existing GIS framework [12].

To enable accurate 3D visualization of any place on the Earth, our 3D GIS subsystem use ellipsoid based Earth model. Different terrain levels of detail (LOD) are dynamically created based on the observer's altitude and position. Each LOD consists of several blocks which has assigned texture and digital elevation model (DEM). Beside terrain visualization, the application can automatically create and visualize 3D objects (e.g. buildings) based on a 2D basis and maximum height. An illustration of a scene appearance constructed with the 3D GIS is shown in Fig. 4.

Camera video registration into such 3D GIS scene is based on previously described method. Fig. 5 illustrates registration results viewed from the camera's position, while Fig. 6 represents a view to an augmented virtual environment from an arbitrary position. More comprehensive video demonstration of *GeoScopeAVE* can be found at http://www.youtube.com/watch?v=8S-MhtEj2O0.

From the Fig. 6 it is clearly seen that a system that would use this approach is highly dependent on underlying GIS model. Since we are using simplified buildings representation, which does not include roof models; we have biggest registration errors in those areas. Nevertheless, when the scene is viewed from the camera position (or some near position) these registration errors seams less noticeable.

Without neglecting of identified shortcomings the implemented prototype proved the applicability of the proposed method.

Fig. 4. 3D GIS scene appearance

Fig. 5. Registration of camera video into 3D GIS

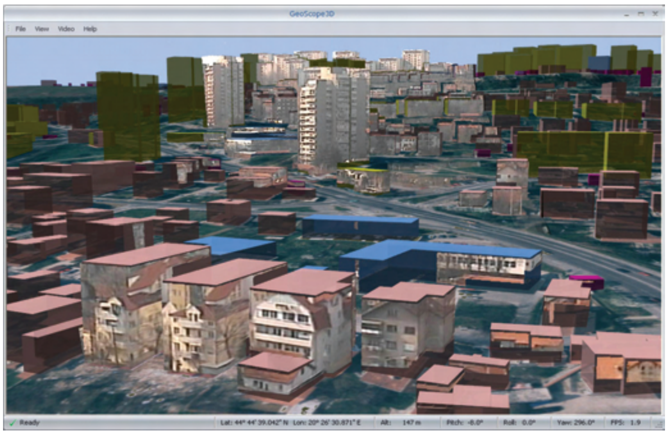

Fig. 6. An arbitrary view to an augmented virtual environment

IV. Conclusions

In this paper, we have presented the method for integration of GIS and video surveillance using augmented virtual environments. To enable fusion of surveillance camera video with 3D GIS virtual environment, we proposed an appropriate PTZ camera view model and registration method. In order to test proposed method we implemented *GeoScopeAVE* prototype which was described in the paper.

The major benefit from the presented approach comes from the fact that camera video is integrated within virtual scene and can be viewed from an arbitrary point and direction. It also has a potential of integration of multiple camera views into single virtual environment. Our future work will go in that direction.

However, there are also some serious drawbacks that should be stated. A system that would use this approach is highly dependent on underlying GIS model. It is also clear that objects (i.e. cars, trees, lamp poles) that are not present in the model are projected onto the buildings or the terrain and may look warped and distorted from other viewpoints. Another issue with AVE approach is performance. Calculation of camera visible surfaces, projection and update of frame textures are both compute and data–intensive.

The presented approach has potential applications beyond video surveillance. For example, it can be used for texture generation, validation, and construction of 3D models.

References

[1] K. Chang, *Introduction to Geographic Information Systems. 3rd ed.*, New York, NY: McGraw–Hill, 2005.

[2] S. Brooks and J.L. Whalley, “Multilayer hybrid visualization to support 3D GIS”, Computers, Environment and Urban Systems, vol. 32, no. 4, 2008, pp. 278–292.

[3] D. Li, Q. Zhu, Q. Liu and P. Xu, “From 2D and 3D GIS for CyberCity”, Geo–Spatial Information Science, vol. 7, no. 1, 2004, pp. 1–5.

[4] I. Oner Sebe, J. Hu, S. You and U. Neumann, “3D Video Surveillance with Augmented Virtual Environments”, First ACM SIGMM international workshop on Video surveillance, 2–8 November 2003, Berkeley, CA, pp. 107–112.

[5] N. Kawasaki and Y. Takai, “Video Monitoring System for Security Surveillance based on Augmented Reality”, Proceedings of the 12th International Conference on Artificial Reality and Telexistence, 4–6 December 2002, Tokyo, Japan, pp. 180–181.

[6] K. Sankaranarayanan and J.W. Davis, “A Fast Linear Registration Framework for Multi–Camera GIS Coordination”, Proceedings of the 5th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS'08), 1–3 September 2008, Santa Fe, USA, pp. 245–251.

[7] U. Neumann, S. You, J. Hu, B. Jiang, J.W. Lee, “Augmented Virtual Environments (AVE): Dynamic Fusion of Imagery and 3D Models”, Proceedings of the IEEE Virtual Reality 2003, 22–26 March 2003, Los Angeles, USA, pp. 61–71.

[8] P. Milgram and F. Kishino, “A Taxonomy of Mixed Reality Visual Displays”, IEICE Transactions on Information Systems, vol. E77–D, no. 12, 1994, pp. 1321–1329.

[9] A. Milosavljević, A. Dimitrijević and D. Rančić, “GIS augmented video surveillance”, International Journal of Geographic Information Science, accepted for publication.

[10] J. Titus, “Make Sense of Lens Specs”, Test & Measurement World, 2005. http://www.tmworld.com/article/CA6252635.html

[11] C. Everitt, “Projective Texture Mapping”, 2001. http://developer.nvidia.com/object/Projective_Texture_Mapping.html

[12] A. Milosavljević, S. Đorđević-Kajan, L. Stoimenov, “An Application Framework for Rapid Development of Web based GIS: GinisWeb”, In: J.T. Sample, K. Shaw, S. Tu, M. Abdelguerfim eds. *Geospatial Services and Applications for the Internet*, New York, NY: Springer, 2008, pp. 49–72.