DYNAMIC OBJECTS MODELING AND 3D VISUALIZATION

Ismail Oner Sebe, PhD. Student
Suya You, Research Assistant Professor
Ulrich Neumann, Associate Professor and Director
Integrated Media Systems Center
University of Southern California
Los Angeles, CA, 90089-0781
iosebe@graphics.usc.edu
suyay@graphics.usc.edu
uneumann@graphics.usc.edu

ABSTRACT

This paper describes an automated approach for dynamic object modeling and visualization of a 3D environment. Moving objects are detected automatically and represented with video textures projected on their convex hulls. These objects are tracked using a winner-takes-all methodology. To achieve robust tracking, a new matching criterion that considers both spatial and temporal coherences of objects is presented. We demonstrate the effectiveness of using this new approach to dynamic shape modeling and representation for creating 3D visualizations of dynamic events observed in real environments.

INTRODUCTION

Civil and defense applications often require and employ 3D models of operational areas. In many cases 3D models of dynamic objects or events are of particular value since they facilitate surveillance, response, and real-time situational awareness in highly-occluded urban settings. The personnel who simulate, monitor, and execute responses to natural or man-made events can gain insights and make better decisions if they have a comprehensive 3D view of the structures and activities occurring at an operational scene. Our goal is to create a 3D "world in miniature" that accurately models the 3D structures and dynamic events of a real world area of interest.

This paper focuses on algorithm for automatic analysis of imagery to track, model, and visualize dynamic scene elements such as people or vehicles at a scene. This work is motivated by our recent research on multi-sensor data fusion and visualization. An Augmented Virtual Environment (AVE), is a novel framework for fusing a number of information sources (e.g., video, image, graphics, text, maps) into a single 3D representation of a scene. We believe that people can best comprehend complex spatial information when it is presented in a common spatial context that mimics the real world [Neumann, 2003]. An AVE system uses a geometric scene model as a 3D substrate for projecting multiple images gathered by fixed or moving image sensors. The resulting visualization is that of a world-in-miniature that depicts the corresponding real world scene and dynamic activities, providing observers with a natural browsing and multi-resolution view of the dynamic spatio-temporal data provided by sensors distributed over a wide environment.

Figure 1 shows an AVE scenario of a campus building complex with an aerial photograph and three video streams projected onto the model. In this aerial view, three moving cameras are depicted by their red wireframe viewing frustums to show their current positions and orientations in the world, and their live video data are projected onto the facades of the buildings in real-time. Note that the visualization viewpoint is completely arbitrary, and the aerial view enables users to comprehend both the camera images and their relationships to the scene.

In many cases, however, simple video projection onto the model does not properly display dynamic objects such as walking people or vechile that are not part of the substrate model. Moving objects appear distorted and misinterpreted when simply projected onto the buildings, as show in Figure 4 (a). For example, a top-view camera will capture a person walking in front of the building from a near vertical aspect. A projection of that video will look realistic when viewed from an aerial viewpoint, however, if viewed from a ground-level viewpoint, the person looks strangely flat and distorted since the figure has no

height in the model and the person's image is simply painted on the street surface. Such distortions and artifacts arise since images lack depth information.

This paper addresses this technical barrier and pursues solutions for it. We present new improvements to our previous work [Sebe, 2003], including a robust dynamic object detection and tracking method; convex-hull shape modeling; and dynamic model visualization. We demonstrate the effectiveness of our new shape model for greatly improving the perception of dynamic events in urban areas.

Figure 1: An AVE system showing projections of three video streams (wire-frame frustums) and an aerial photograph within the USC campus area.

RELATED WORK

Several recent systems address the problems of multiple sensor fusion and data analysis. Distributed Interactive Video Array (DIVA) developed at the University of California at San Diego [Hall, 2002] employs multiple videos, images, and 2D maps for monitoring remote scenes. The system has the capability for visualizing multiple 2D data sources, but does not address 3D visualization so the occlusions that occur in urban areas remain problematic. Spann and Kaufmann (at BAE Systems) developed a system that fuse multiple aerial and/or ground images on 3D terrain models [Spann, 2000], but the system lacks video capability and dynamic object detection. The VideoFlashlight system developed at Sarnoff Corporation provides visualizations of urban sites by fusing multiple video streams and 3D models [Kumar, 2000], but does not address dynamic modeling of moving objects. The Video Surveillance and Monitoring (VSAM) project, conducted at CMU and other institutions developed automated video understanding technologies, enabling a single human operator to monitor activities over a broad battlefield using a distributed network of active video sensors [Kanade, 1998]. The key idea of the VSAM system is to automatically detect and track people, vehicles, and their interactions from multiple video streams, and then insert selected images of interest into an environment map to reduce the operator's cognitive load. Recently, a project called Virtual Soccer Match developed at Keio University, Japan, is able to fuse multiple video sequences in real time [Inamoto, 2003]. Dynamic objects (soccer players) are detected and visualized as pseudo-3D models. This system is specific to soccer field layout – a planar ground model is assumed and multiple cameras are required to reconstruct 3D scene models. While the above systems offer many useful capabilities, they lack the ability to fuse multiple video streams, images, data, and dynamic models as 3D elements in a common 3D scene model.

DYNAMIC OBJECT DETECTION

Assuming stationary cameras are used, our goal is a robust detection of moving objects under a variety of conditions including sensor noise, low-resolution, and varying illumination. A trained background-subtraction approach is employed to segment the moving objects from background. This method is suitable for stationary cameras where a relatively long sequence of video (in the order of minutes) is always available for background learning [Harville, 2001]. Background learning is an important step in a segmentation approach. There are several important factors that a background learning algorithm should deal with in order to achieve a robust segmentation, such as illumination change, dynamic background, high-traffic problems, camouflage, and occlusions. These problems have been extensively investigated, but better approach still needs to be pursued. Simple background averaging or estimation by per-pixel

Gaussian distribution is popular methods, but often fail to model higher order backgrounds (e.g. bi-modal backgrounds). A per-pixel Kalman filter was also proposed for background estimation [Ridder, 1995], but its results have similar characteristics as that of the Gaussian distribution method. To model complex multi-backgrounds, the Gaussian mixture model was suggested by Stauffer and Grimson [Stauffer, 1999]. This method is able to prevent detection of bi-modal movements such as waving trees or flickering monitors, but suffers from the problems of the complex parameters tuning and implementation. An improved approach was investigated in [Harville, 2001] by considered depth information for background learning. In [Toyama, 1999] a method called Wallflower is proposed by incorporating the multiple tasks of pixel classification, region detection, and tracking to an integrated solution to handle complex background estimation. An extensive survey of various background learning models is also given in this paper.

The accuracy of background estimation directly determines the performance of a dynamic object detection system. We proposed to use an adaptive variable-length time averaging algorithm to estimate background image. A sliding window is used for time averaging to dynamically model a single distribution background. Our experiments show that the method offers performance that is similar to that of a single Gaussian distribution. However, the time averaging method typically requires a big memory buffer to be allocated for frame averaging within a sliding window, i.e. the size of the image buffer equals the product of the time-window length and the image size. For example, a 256x256 size image using a length of 1500 time-window will result in 73.7M memory being allocated! To track this problem, we developed a fast, low complexity, and low storage variable-length time averaging method. In our method, only 2 image buffers are required: one for the actual background image, and one for temporary buffer. We start with an empty temporary buffer and at every frame, add current frame to this buffer. We need N (typical $N = 1500$) frames for background estimation. Since we only have two image buffers, we do not have to store any additional frames. Once all N frames are added, we transfer the temporary buffer to background buffer by dividing each pixel value by N. This division is performed to keep the pixel values between 0 and 255. After the data transfer, the temporary buffer is initialized again to zero. During the next N frames, we use current background image and estimate a new one with the temporary buffer. In short, the background image is updated at every N frames with an average of the most current N frames.

Background estimation is followed by a pixel subtraction processing to segment the foreground objects. Difference image is estimated by taking the absolute difference of the current frame and the estimated background image:

$$\Delta I(x, y) = \left| I(x, y) - B(x, y) \right|, \tag{1}$$

where $\Delta\mathbf{I}$ is the difference image, $\mathbf{I}$ is current frame, and $\mathbf{B}$ is the estimated background image.

The difference image is then binarized using a histogram-based threshold process. A histogram of the difference image is calculated and converted into a cumulative distribution function used to for threshold estimation (i.e. HISTOGRAM_THRESHOLD). The threshold could be estimated by simply setting to it a certain value so that only the top 5% differences are labeled as foreground objects. However, using this method alone is not robust, since the method returns a value for threshold regardless of the existence of a moving object in the scene. In the case of no objects, noise is detected as foreground. We therefore propose to use equation (2) to determine the segmentation threshold.

$$Threshold = \max(MIN\_THRESHOLD, HISTOGRAM\_THRESHOLD), \tag{2}$$

,where MIN_THRESHOLD is a fixed lower bound (we use a value of 25/255).

The difference image is filtered with the estimated threshold: a pixel having larger value than this threshold is classified as foreground, otherwise classified as background.

Finally, a morphological filter [Jain, 1995] is operated on the segmented binary image to remove too small foreground areas, and then a two-pass 4-neigbors connectivity algorithm is used to label the segmented areas as moving objects or background [Gonzales, 1993].

DYNAMIC OBJECT MODELING

The output of the object detection module is a labeling list of detected objects containing the object IDs and pixel locations. Although this type of object representation is simple to use, it doesn't contain any simple shape information. We need to model them using a more compact graphic representation. In our previous work, we use a dynamic single rectangular polygon to approximate the model of detected moving object which produce more realistic and less distorted AVE visualization [Sebe, 2003]. However, the rectangular model can not accurately describe the shape of the object, resulting in noticeable artifacts. In this paper, we propose to use 2D convex-hull representation to model the moving objects. We feel that this new representation can be more realistic to describe a range of object shapes such as vehicles and people, achieving a comfortable visualization. There are several methods to approximate or find convex hull of a dataset. In our work, we employ a fast model fitting method proposed in [Andrew, 1979]. This method has computational complexity of O(n log n) and only requires an easy presorting on the input data [Sunday 2001].

(a) (b)

Figure 2: a moving vehicle is detected and modeled as a convex hull with 18 points (a), and a rectangular polygon (b)

A convex hull representation is capable of capturing the spatial shape property of an object. The representation is compact, needing only few parameters to control the shape variations. Besides these spatial control parameters, in our work, we also estimate inter-frame object motions, and use the estimated motion vectors as temporal coherence descriptor of the moving objects. The motion estimation is done by correlation matching (the sum of squared differences) between convex hulls of detected objects in neighboring frames. A 5x5 search window is used in our current implementation. For every detected object, a motion vector is estimated and used for robust object tracking. Figure 2 (a) shows the results of using above approaches to detect and model a moving vehicle. In this example, the detected vehicle is represented by 18 convex points. For visual comparison, we also show the result of using a single rectangular polygon to model the moving vehicle Figure 2 (b).

DYNAMIC OBJECT TRACKING

The detected moving objects in the scene are interpreted as belonging to one of the possible states: appear, disappear, or track. To do so, an object history is created and updated during the tracking procedure. To decide which state is more appropriate for a particular object, a matching criterion is required. A robust matching algorithm should account for both spatial coherence and temporal coherence. In our work, the spatial coherence criterion is defined as a function of overlap of current object with the known objects (object history) and their corresponding sizes, which is defined as:

$$SizeMatch = \frac{Int\ (O_1, O_2) * \sqrt{S_1 * S_2}}{(S_1 + S_2)/2}, \tag{3}$$

where S_1 and S_2 are the size of objects O_1 and O_2, respectively, and '**Int()**' is a function that returns the size of the overlapping of the two objects. The spatial matching criterion contains three components: *temporal coherence* of objects (the size of overlap); *geometric mean* of objects (numerator); and *arithmetic mean* of objects (denominator). Note that the ratio of the *geometric mean* and the *arithmetic mean* measures the confidence of shape matching of the two objects. Figure 3 shows the behavior of this ratio for different values of S_1 and S_2.

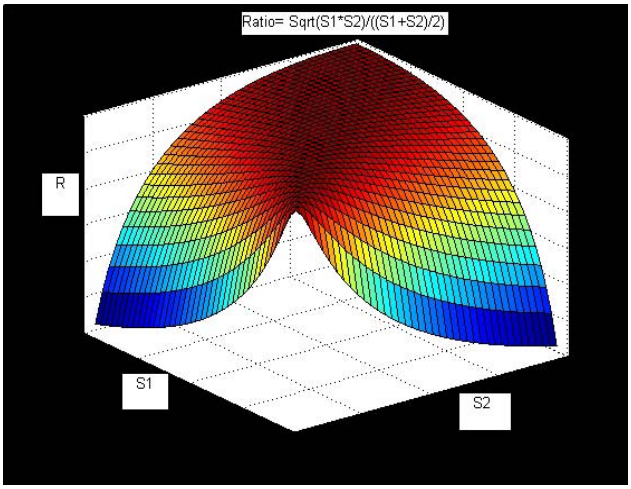

Figure 3: Ratio plot defined in equation (3) without the overlap parameter. S_1 and S_2 are varied and maximum occurs when S_1=S_2. The plot has a maximum of 1 (Red in color) and lowest value of 0 (blue in color).

The overlap function **Int()** has an upper limit of min(S_1, S_2). Although the overlap of two objects can be found by intersecting the two convex polygons, this procedure is too costly for a value which itself is an approximation. To simplify computation, we use the object's single polygon representation (i.e. a rectangle bounding the detected object) to compute the overlap. Its matching result is very similar to that of a convex hull intersection.

The motion vector that measures the temporal coherence of moving objects is also helpful for match tracking. The cosine of the angle between two vectors measures their direction agreement. (The Cosine is scaled to [0, 1] to simplify calculations.) Considering both the direction and magnitude of vectors, we create the following temporal coherence criterion for motion matching:

$$MotionMatch = \frac{\sqrt{\|V_1\| * \|V_2\|} * (Cos(\Theta) + 1)/2}{(\|V_1\| + \|V_2\|)/2}, \tag{4}$$

where $\|V_1\|$ and $\|V_2\|$ are the magnitude of the motion vectors, and Θ is the angle between the two vectors. This cost function has a range of [0, 1]. The maximum value of 1 is achieved when V_1=V_2. However, there is a singular case if we directly use equation (4). Since the motion vector is estimated between two consecutive frames, slowly moving objects may produce zero motion vectors that result in an undefined match. To handle this case, we modified equation (4) to obtain the following motion matching criterion:

$$MotionMatch = \begin{cases} 0.5 & \text{, if } (V_1 = 0 \text{ and } V_2 \neq 0) \text{ or } (V_1 \neq 0 \text{ and } V_2 = 0) \\ \dfrac{\sqrt{\|V_1\| * \|V_2\|} * (Cos(\Theta) + 1)/2}{(\|V_1\| + \|V_2\|)/2} & \text{, if } V_1 \neq 0 \text{ and } V_2 \neq 0 \\ 1 & \text{, if } V_1 = V_2 = 0 \end{cases}, \quad (5)$$

Considering both spatial coherence and temporal coherence, the overall motion matching criterion is defined as:

$$Match = Int(O_1, O_2)\left(\frac{\sqrt{S_1 * S_2}}{(S_1 + S_2)/2} + \lambda \begin{cases} 0.5 & \text{if } (V_1 = 0, V_2 \neq 0) \text{ or } (V_1 \neq 0, V_2 = 0) \\ \dfrac{\sqrt{\|V_1\| * \|V_2\|} * (Cos(\Theta) + 1)/2}{(\|V_1\| + \|V_2\|)/2} & \text{if } V_1 \neq 0 \text{ and } V_2 \neq 0 \\ 1 & \text{if } V_1 = V_2 = 0 \end{cases} \right), \quad (6)$$

where λ is a weighting constant. $\lambda = 0.5$ is used in our experiments.

The matching criterion is computed between every new detected object and a recent history of objects, resulting in a matching table of size *M* by *N*, where *M* is the number of history objects, and *N* is the number of new detected objects. We match objects and assign appropriate object IDs using this matching table and a winner-takes-all mechanism. The best match in the whole matching table is assigned first, and then both the object and its best history match are removed from the table. This procedure is repeated until all objects are assigned. For new objects that appear, the table will produce a zero row. These objects are assigned new IDs and added to the object history.

DYNAMIC OBJECT VISUALIZATION

The tracked objects need to be placed into the 3D scene model for AVE visualization. We use the assumption that tracked objects (people and vehicles) rest on the ground. Objects are rendered as 2D planar convex polygons in 3D world. The mid-point coordinate of the lower edge of the bounding box of a tracked object defines its contact point on the ground. A ray from the camera viewpoint through the contact point in the image is intersected with the ground model to determine the 3D position for a dynamic pseudo-3D model of the moving object.

To illustrate the effectiveness of our new dynamic model representation, Figure 4 shows a comparison of three ways to visualize live video streams containing moving vehicles and people. Figure 4 (a) illustrates the image projection of a moving car without using any corresponding model. The video cameras are at ground level so their projections of a car or a person appear distorted since there are no models of these moving objects in the 3D scene model. The person and car appear to be "smeared" over the road and part of the building when viewed from our raised viewpoint. Figure 4 (b) shows the result of applying the modeling approach in [Sebe, 2003] to the same scene. In this result, the moving car is tracked and modeled using a rectangular polygon. We can note that the rectangular model does not accurately capture the shape of the object, resulting in noticeable artifacts. Figure 4 (c) is the result of using the approach in this paper. The new model representation more accurately captures the shape of the moving vehicle, improving the display of video textures projected onto the model. Figure 5 shows different results of the new modeling system. People, cars, and SUVs are modeled and visualized without any human intervention. Our current AVE display system achieves real time performance (~25fr/sec) on a 2.2GHz Pentium-4 workstation, including all the computations for using 3 live firewire cameras, one high resolution aerial photo, and a 3D model of the entire USC campus and surround areas (over 200 buildings).

Figure 4: (a) An image projection of a moving car without a dynamic model results in a distorted presentation, (b) the moving car is modeled as a rectangular polygon, and (c) is modeled as a convex polygon.

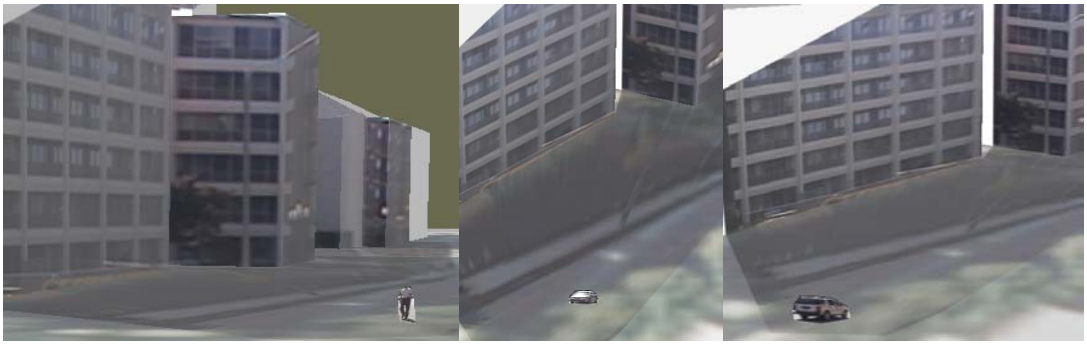

Figure 5: Visualization results using the method proposed in this paper: (a) a human walking, (b) a small car, and (c) an SUV.

CONCLUSION

We present a novel visualization system suited for visualizing 3D moving objects in large-scale urban areas. The AVE system fuses multiple data sources, such as LIDAR, images, and video. Dynamic object detection, modeling, and tracking enable our system to generate arbitrary views of both static and dynamic objects. A convex hull representation is computed for geometric modeling and motion vectors are used for temporal modeling of the detected objects. Furthermore, we developed a matching criterion that uses size, shape, location, and movement of the objects all at once. A winner-takes-all object assignment procedure is used to track multiple objects in time and space. Future work is required for multi-object tracking. Splitting and merging of the objects is not supported in our current method.

ACKNOWLEDGEMENT

This work was supported by the National Geospatial Intelligence Agency (NGA) under a NGA University Research Initiative (NURI) program. We thank the Integrated Media Systems Center, a National Science Foundation Engineering Research Center, for their support and facilities. We thank the Airborne1 Inc. for providing us with the USC campus LiDAR data. Our thanks also go to HP, Intel, and Microsoft for equipment donations.

REFERENCES

Andrew, A.M. (1979). Another Efficient Algorithm for Convex Hulls in Two Dimensions. Info. Proc. Letters. 9, 216-219.

Gonzales, R.C., R.E. Woods (1993). Digital Image Processing. Addison-Wesley. 2nd edition. September 1993. pp 41-44.

Hall B., M. Trivedi (2002). A novel graphical interface and context aware map for incident detection and monitoring. 9th World Congress on Intelligent Transport Systems. October 2002.

Harville, M., G. Gordon, J. Woodfill (2001). Foreground Segmentation using Adaptive Mixture Models in Color and Depth. Proceedings of the IEEE Workshop on Detection and Recognition of Events in Video. July 2001.

Inamoto, N., H. Saito (2003). Immersive Observation of Virtualized Soccer Match at Real Stadium Model. ISMAR 2003. pp 188-197.

Jain, R., R. Kasturi, B.G. Schunck (1995). Machine Vision. McGraw-Hill International Edition. pp 76.

Kanade, T., R. Collins, A. Lipton, P. Burt and L. Wixson (1998). Advances in cooperative multi-sensor video surveillance. Proc. of DARPA Image Understanding Workshop. Vol. 1, pp. 3-24.

Kumar R., H.S. Sawhney, Y. Guo, S. Hsu, S. Samarasekera (2000). 3D manipulation of motion imagery. ICIP2000. September 2000.

Neumann, U., S. You, J. Hu, B. Jiang, and J. Lee (2003). Augmented Virtual Environments (AVE): Dynamic Fusion of Imagery and 3D Models. IEEE Virtual Reality '03. Los Angeles, CA. March 2003.

Ridder, C., O. Munklet, H. Kirchner (1995). Adaptive background estimation and foreground detection using Kalman-filtering. ICRAM'95. pp. 193-199.

Sebe, I.O., J. Hu, S. You, U. Neumann (2003). 3D Video Surveillance with Augmented Virtual Environments. ACM Multimedia 2003 Workshop on Video Surveillance. Berkeley, CA. November 2003.

Segal, M., C. Korobkin, R. Van Widenfelt, J. Foran, and P. Haeberli (1992). Fast shadows and lighting effects using texture mapping. SIGGRAPH '92. pp. 249–252.

Spann J.R., K.S. Kaufman (2000). Photogrammetry using 3D graphics and projective textures. IAPRS 2000, Vol. 33.

Stauffer, C., W.E.L . Grimson (1999). Adaptive background mixture models for real-time tracking. CVPR 99. Vol. 2. pp. 246-252. June 1999.

Sunday, D. (2001). http://www.softsurfer.com/Archive/algorithm_0109/algorithm_0109.htm.

Toyama, K., J. Krumm, B. Brumitt, B. Meyers (1999). Wallflower: Principles and Practice of Background Maintenance. ICCV 1999. pp 255-261.