3D Scene Reconstruction System from Multiple Synchronized Video Images

Taewoo Han[1], Juho Lee[2], Hyung S. Yang[3]
AIM Lab., EE/CS Dept., KAIST[1,2,3]
373-1, Guseong-dong, Yuseong-gu, Daejon, Republic of Korea
{ bluebird[1], jhlee[2], hsyang[3] }@paradise.kaist.ac.kr

Abstract

One of the goals of three-dimensional (3D) computer graphics is to create virtual realistic images of dynamically changing scenes. In this paper, we designed and implemented a system that processes multiple synchronized video sequences and generates 3D rendering of dynamic objects in real time. The video-image-based virtual volumetric scene reconstruction system acquires synchronized multiple video images and renders dynamic real-world scenes. It includes an efficient image-based reconstruction scheme that computes and shades 3D objects from silhouette images, as well as it includes a silhouette extraction scheme that is robust to illumination change. The proposed system is relatively low-cost and does not require foregoing any special hardware or specific environment.

Key words: scene reconstruction, visual hull, image-based rendering

1. Introduction

In computer graphics and computer vision, volumetric scene reconstruction of a 3D model from multiple two-dimensional (2D) photographic images is already an old and one of the most important problems. It remains one of the most difficult problems. But, many researchers have been working on the creation of virtual scenes from images in many applications, such as virtual reality, games, multimedia, robot navigation, and special effects for moving pictures [1].

Generally, the volumetric structure of a scene can be reconstructed if its material characteristics, illumination, and geometric constraints are carefully considered. The methods used to acquire 3D information on dynamically changing scenes are classified into two approaches, e.g., the active and the passive method according to the types of imaging sensors.

Active approaches use structured lights or laser scanners to directly acquire 3D information about the subject [2]. They produce high quality data since the emitted lasers or lights directly obtain the range to the parts of the subject. However, the equipment used in active approaches is almost considerably expensive and have physical restrictions such as particular light and special painting. As well, they are not adequate to capture dynamically changing scenes in real-time as it takes a long time to get data for one scene.

Compared to active approaches, passive approaches extract 3D data indirectly without contact with the object. Utilizing the picture images taken from cameras is representative of passive approaches [1][3][4]. Traditionally extracting 3D data using passive imaging is less accurate than using expensive active sensors. Also it needs dramatic computing time.

The most ideal system must be able to construct high-quality images in a short time using low-cost equipment. Moreover, it should not be restricted to any environment and should have a broad field of application that encompasses even such areas as sports, dance, and remote video-conferencing.

In this paper, we designed and implemented a system that acquires synchronized multiple video images and reconstructs virtual scenes cost-effectively using the silhouette images. The system should be capable of real-time synchronous capturing of camera video images, camera calibration, and silhouette extraction that is invariant to illumination change. It computes and shades 3D objects using the image-based visual hull. It contributes to the speed-up and quality improvement of the previous reconstruction methods. In the next chapter, we roughly describe techniques to reconstruct scenes from photographs. Then we explain the system's algorithm. In Chapter 4, we show the designed system structure and the experimental results. And we conclude this paper in Chapter 5.

2. Virtual Scene Reconstruction Techniques

There are many methods for 3D scene reconstruction based on passive imaging. The active methods require special hardware or a specific environment. But passive methods do not need the special hardware except some camera and capture board. The passive methods are divided into voxel coloring, stereo vision, image-based rendering, and visual hull.

2-1. Voxel coloring

Voxel coloring depends on color consistency [5-8]. If the colors from the different cameras are the same at a visible point in 3D space, that point exists. Otherwise, that point does not exist. The camera, illumination, and other external conditions may affect color consistency accordingly, thereby yielding incorrect results.

2-2. Stereo vision

Stereo matching is used to obtain the range information from a pair of 2D images. In this method, the robust search for the same point in two images is very difficult [9-10]. As such, the camera views are often arranged along the baseline, and in most cases, assuming a limited disparity range. Another limitation of the stereo matching method is the occlusion problem.

2-3. Image-based rendering

Image-based rendering is another method of modeling and rendering [11-13]. The key advantage of this technology is its realistic results. By synthesizing the resulting image directly from images without the traditional modelling process, we can obtain the resulting image regardless of the geometric complexity of the object and the complexity of the image. However the depth is almost flat.

2-4. Visual hull

The visual hull is defined as a maximal volumetric shape that makes the same silhouette from all views of the real object [14]. It is similar to the convex hull, although it can have holes. When many cameras are used, the inferred visual hull approximates the original shape of the object. However the obtained visual hull does not correspond to the original shape of the object because of its concave regions [15-16].

3. System Algorithms

3.1 Algorithm design objective

The first system developed in this paper was a trial of the method, using the stereo vision that is similar to Kanade's method, to determine the evaluation factor of system design and algorithm selection. We made an effort to reconstruct the object in 3D virtual space by combining several depth maps calculated from feature disparities in captured real scenes. With this system, we found out that disparity is not robust to texture, illumination, and background. Moreover stereo-based 3D reconstruction is expensive, and it is also difficult to combine the resulting depth maps such as zippering.

Through developing this system, we defined the design criteria for implementation. First, the system should not require any special hardware or specific background to acquire the real scene, e.g., special illumination equipment such as that which emits the grid lights or the laser range scanner should be unused. Second, the algorithm should be so handy that special hardware would not be required to process the reconstruction. Special machines such as digital signal processors (DSPs) or a distributed computing environment using multiple processors or heavy workstations should not be assumed. Third, implementation should produce the result continuously within a low latency time after acquiring the input images. In the end, the system should be real-time. Fourth, the resulting scene should be as realistic as possible.

For this paper we selected the passive acquisition method to satisfy the first condition, and the visual hull method to satisfy the second and the third conditions. Furthermore, we used the image-based rendering technique to satisfy the fourth condition.

In addition, we used the image-based visual hull to execute image-based rendering, with the intersection of rays rather than in the voxel space to achieve the cost-effective results.

3-2. Visual hull sampling

In this paper, a virtual image reconstruction scheme based on the silhouette image from each camera is used to reconstruct the object based on the visual hull of 3D virtual space (Figure 1). After polyhedra are generated from the center of projection (COP) of the camera and from the silhouette image corresponding to that camera, the visual hull of the object is obtained by intersecting these polyhedra. In addition, the range image is obtained by projecting the rays from the COP towards the object at equal spacing after assuming that a virtual camera exists. This process is called visual hull sampling.

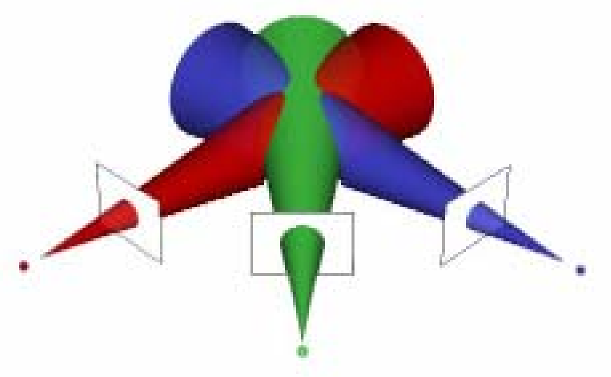

Figure 1 Visual hull: Visual hull is defined by intersection of cone-shaped polyhedra from extrusion of image silhouettes.

When the calibration information of the cameras to be used in capturing scenes as well as the captured images are given, we can calculate the range image of the virtual camera.

In this calculation, the projective geometry is very important. We only consider the pinhole camera model. Let (p_x, p_y) be a center of image coordinates, f be a focal length, *dpx* and *dpy* be x, y directional lengths corresponding to one pixel size, then projective transform of a point $X_c = [x, y, z, 1]^T$ in 3D homogeneous space is written as

$$U = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f \times s / dpx & 0 & p_x & 0 \\ 0 & f / dpy & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = K[I|0]X_c \qquad (1)$$

In the above equation K is called the camera matrix. Since X_c is a point in the camera coordinate system, X_c needs to be transformed to a point in the world coordinate system. When the rotation matrix of a camera is R and a vector C represents the center of the projection (COP) of the camera, this transform is expressed as followings:

$$X_c = \begin{bmatrix} R & -RC \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} R & -RC \\ 0 & 1 \end{bmatrix} X \qquad (2)$$

Substituting X_c in the equation (2) for X_c in the equation (1) we can get the following equation.

$$U = K[I|0] \begin{bmatrix} R & -RC \\ 0 & 1 \end{bmatrix} X = K[IR| - RC]X = KR[I| - C]X \quad (3)$$

If we express X in the non-homogeneous coordinate form, that is $\hat{X} = [x, y, z]^T$, the equation (3) can be written as

$$U = (KR)(\hat{X} - C) = P^{-1}(\hat{X} - C) \qquad (4)$$

where P is the transformation from image coordinates $U = [u, v, 1]^T$ to a line in 3D space. So, the image coordinates corresponding to the 3D point can be computed by the equation (4). And a line through two points, the COP of a camera and an image point U, is expressed by

$$X(t) = C + tPU \qquad (5)$$

where t is a parameterization variable for the line and is used to describe a particular 3D position on the line.

If we carefully consider the relationship between rays of one camera and an image space of the other camera, we will realize that many 3D rays from one camera are projected to a single line in image space of the other camera. Figure 2 shows an example of this relationship. The single line is called an epipolar line. Fundamental matrix depicted in the equation (6) relates a pixel of one camera image to an epipolar line.

$$F = \begin{bmatrix} 0 & -e'_z & e'_y \\ e'_z & 0 & -e'_x \\ -e'_y & e'_x & 0 \end{bmatrix} P'^{-1} P \qquad (6)$$

where coordinate $[e'_x, e'_y, e'_z]^T$ is called the epipole, the projection of the COP of one camera to the image space of the other camera. Using the fundamental matrix we can compute the coefficients, a, b, and c of the line, $ax + by + c = 0$

We compute the silhouette image for each image, and subsequently initialize the virtual range set image to the interval from 0 to infinity. A pixel of the virtual range set image represents the interval wherein the object may exist. After that interval is determined for each image, the virtual range set image is reduced via the set intersection operation.

The process for computing the virtual range set image follows. First, we calculate the 3D ray from the calibration information of the virtual camera. We then project that ray onto the image space of one real camera to get the projected line. Next we find the intersection points of the line with the silhouette edges. We calculate the recovered 3D-line intervals by reprojecting the intersection points into the 3D space. Finally, we find the intersection intervals of the virtual range set image with the recovered 3D line intervals.

3-3. Performance improvement of the visual hull sampling algorithm

When the virtual scene is reconstructed using the visual hull method, there are various techniques to improve the speed and correctness of the reconstruction algorithm. First, we can utilize the line caching to calculate the intersecting points of the silhouette edges and the projected line. Second, the code optimization strategy can be used to speed up the system.

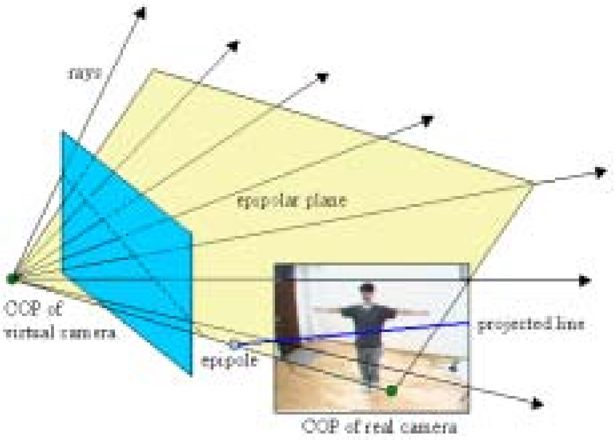

Figure 2 Multiple rays of one camera are projected to the same line in the image plane of another camera.

If we carefully consider the relationship of the camera rays and another image space, we will realize that many 3D rays from the camera are projected to the same line in another image space. This projected line is called the epipolar line. Figure 2 shows an example of this relationship.

Because of this fact, the process of finding the intersection intervals of the projected line and the silhouette edges is redundant. In this process, line caching can be used. When caching is utilized, we first compute the caching index to check if the intersection intervals have already been calculated. In case the cache bucket is already full, we simply use the intervals in it. In case the cache bucket is empty, we find the intersection intervals to save those in the cache bucket. For the cache index, we use the further point from the epipole among the intersection points of the projected line and the boundary of the segmented object (Fig. 3).

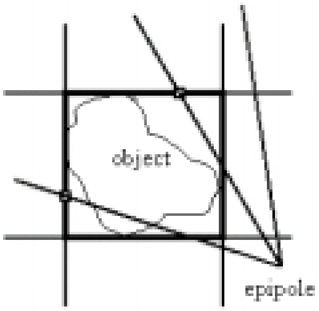

Figure 3 For reducing the computation cost, intersection points of the epipolar line with the rectangular boundary surrounding the object are used as a caching index.

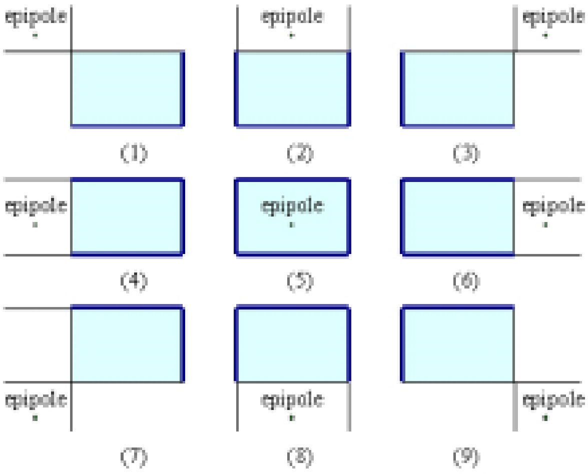

Figure 4 Indexing points are calculated separately for nine cases according to the relative position of the epipole to the rectangular boundary of an object.

Instead of calculating the distances to the two intersection points, we directly find the point that will be used as the cache index through the positional information of the epipole, which will more significantly reduce the computing time. The positional relation of the epipole and the rectangle boundary of the object can be classified into 9 cases, which are shown in Figure 4. In this figure, we use the intersection point of the projected line and the thick boundary as a cache index.

As a second method to speed up the system, code optimization techniques are employed. When the intersection points of the projected line and the silhouette edges are computed, the digital differential analyzer is utilized to reduce the addition and multiplication operation of the real numbers. In this system, we used the floating-point version of the Brensenham's line drawing algorithm because the end points of the projected line are given in real numbers. While casting the real number to an integer number can speed up the system, it may compromise its accuracy.

Static memory can also be allocated instead of dynamic memory to speed up the system. However, dynamic memory is preferred since the intersection intervals in the 2D or 3D space vary in relation to the silhouette image and the virtual camera.

Another optimization technique involves the reduction of the floating-point operation. The distance from the COP of the camera to the object is represented as a real number, in the same way that the position of the COP of the camera and the projection matrix are represented as a real number. By converting these real numbers to integer numbers the system will become to be fast.

Thus, the system became two times faster than before through code optimizations.

4. System Implementation and Results

4-1. System overview

Four sets of general color-image, NTSC-output CCD camera (Samsung SC340, interlaced scan, 1CCD, Bayer-patterned) and 5mm~15mm canon lenses with 4 frame-grabbers are used to capture moving images, while 60Hz AC power is used for synchronization. With only one PC, the system captured 320x240 24-bit images in real time (30Hz) from 4 cameras much more easily than Kanade's system and other systems, with comparable or better performance.

4-2. Silhouette extraction

Generally, the extraction of the silhouette employs background subtraction. Illumination changes and shadows make extraction of the correct silhouette difficult. In this paper, we implemented a mathematical model between the illumination intensity, reflection index of objects, and pixel values of the images [17]. Assuming that the distribution of illumination intensity in a very small region is constant, we define the variables as follows:

- r: reflection index of one point of the object
- α: illumination intensity of one point of the object
- β: bias value of the capturing equipment

The pixel value of the image is defined as: q= αr + β. Here, the change in the illumination intensity affects the pixel value linearly. Thus,

$$p = q - \beta = \alpha r.$$

Therefore, the pixel values of the reference image and the input image as follows:

$$p_i^{(r)} = \overline{p}_i^{(r)} + \Delta p_i^{(r)} = \alpha^{(r)} r_i + \Delta p_i^{(r)},$$
$$p_i^{(i)} = \overline{p}_i^{(i)} + \Delta p_i^{(i)} = \alpha^{(i)} r_i + \Delta p_i^{(i)}.$$

The ratio of illumination is written as:

$$\bar{t} = \frac{\alpha^{(i)}}{\alpha^{(r)}}.$$

Assuming α is not changed, the value t is also constant at the same value r. Using this relationship, the variation of the illumination intensity in the same position according to the illumination change is calculated as shown in Figure 5.

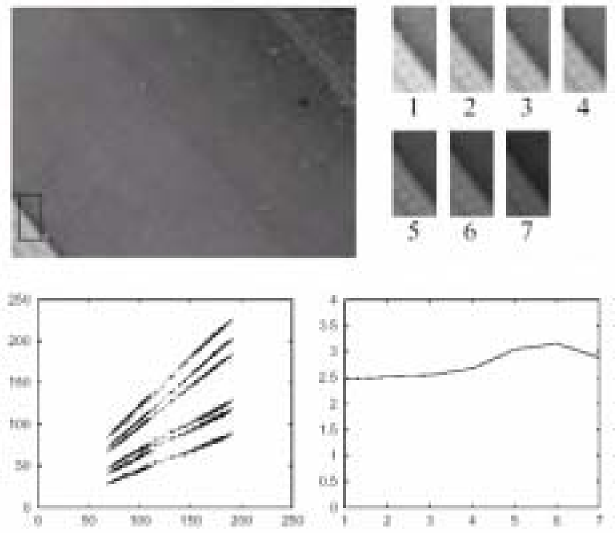

Figure 5 Variation curve of the pixel value (bottom left) and the standard deviation (botom right) according to 7 different illumination conditions of the same position

In the image, the pixel value of the x and y coordinates is written as: (x, y) = (αxr + β, αyr + β). Therefore, we can calculate β using the crossing point of the graph as follows:

Fitted = (-5.747, -4.334),
β =(-5.75-4.33)/2= -5.04, σ^2 = 2.75

We use the standard deviation as the threshold value, when the window size is 7x7 pixel. The boundary of the object that is robust to the illumination condition is consequently obtained by determining the highly changing position.

Figures 6 show the sample of the input image, and Figures 7 and 8 show the resulting image and the segmented image.

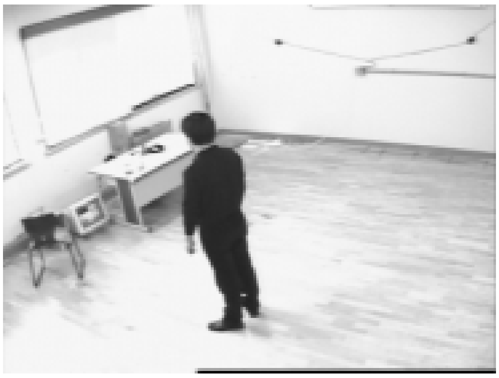

Figure 6 Input image

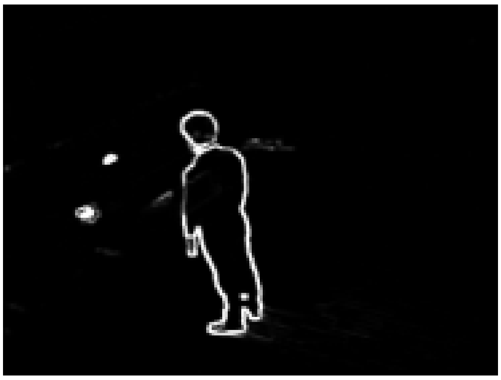

Figure 7 Resulting image

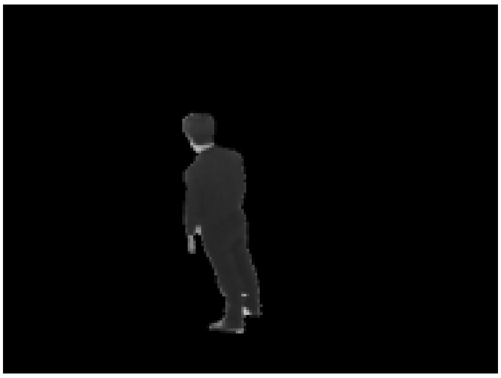

Figure 8 Segmented image

4-3. Virtual scene reconstruction results

From our implementation of the virtual scene reconstruction system, we saw that the system could reconstruct the virtual image at a speed of about 8 frames per second using the 320x320 24-bit color images from four cameras. Figure 9 and 10 show the sample of reconstructed images.

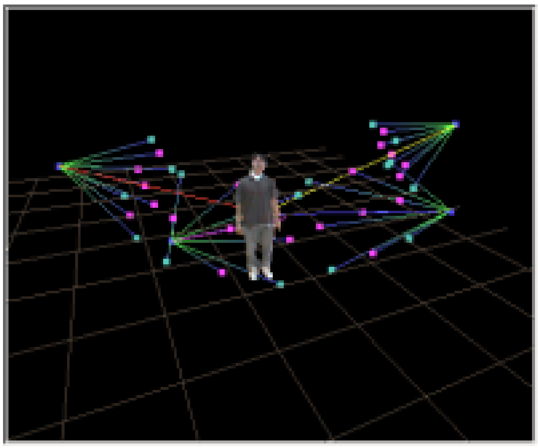

Figure 9 The virtual image 1

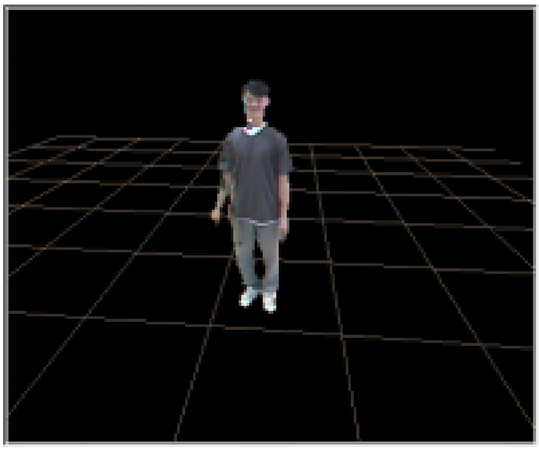

Figure 10 The virtual image 2

5. Conclusion

In this paper, we implemented a system that can reconstruct virtual scenes from real video streams without the need for special hardware, using only one PC with general video capturing components. The system was implemented by computing visual hull range data sampled from a virtual camera, using the camera ray's projection and intersection with the silhouette edges and texturing the computed range image through the image of the camera that was most similar to the ray direction of the virtual camera. The system registered an almost real-time performance from continuous video streams using only an ordinary PC system.

Therefore, the system can be applied to various fields given an improved quality of the generated image and the system's performance. Some Applications of this system include: the real-time remote virtual presence system; digitization of sports, dance, martial arts, etc.; and the virtual camera walking system for cinema production, including 3D object modeling.

Ensuring the quality of the images for use on broadcast and cinema requires high-quality segmentation, more accurate camera calibration, and refinement of the visual hull through stereo vision or voxel coloring. Enhanced image-based rendering techniques may also be tried to improve image quality.

Acknowledgements

The research reported herein was supported in part by Digital Media Lab.

References

1. G. Slabaugh, W. Bruce Culbertson, Thomas Malzbender, and Ron Shafer, "A Survey of Methods for Volumetric Scene Reconstruction from Photographs", International Workshop on Volume Graphics 2001, pp. 81-100, Stony Brook, New York, June 21-22, 2001.

2. Demetri Terzopoulos and Keith Waters, "Modeling and Animating Faces using Scanned Data", the Journal of Visualization and Computer Animation, Vol. 2, pages 123-128, 1991.

3. Gerald Eckert, "Automatic Shape Reconstruction of Rigid 3-D Objects from Multiple Calibrated Images", Proceedings of Eusipco 2000, Tampere, Finland, Sep. 4-8, 2000.

4. Richard Hartley, Andrew Zisserman, "Multiple View Geometry in Computer Vision", Cambridge University Press, 2000.

5. S. Seitz and C. Dyer, "Photorealistic Scene Reconstruction by Voxel Coloring", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1067-1073, June 1997.

6. K. N. Kutulakos and S. M. Seitz, "What Do N Photographs Tell Us about 3D Shape?", TR680, Computer Science Dept., Univ. of Rochester, January 1998.

7. W. B. Culbertson, T. Malzbender, and G. Slabaugh, "Generalized Voxel Coloring", Proceedings of the ICCV Workshop, Vision Algorithms Theory and Practice, Springer-Verlag Lecture Notes in Computer Science 1883, pp.100-115, September 1999.

8. P. Eisert, E. Steinbach, and B. Girod, "Multi-Hypothesis, Volumetric Reconstruction of 3-D Objects from Multiple Calibrated Camera Views", Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, pp.3509-3512, 1999.

9. Y. Yang, A. Yuille, and J. Lu, "Local, Global, and Multilevel Stereo Matching", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 274-279, 1993.

10. Q. Chen and G. Medioni, "A Volumetric Stereo Matching Method: Application to Image-Based Modeling", Proceedings of the IEEE Conference on

Computer Vision and Pattern Recognition, pp.29-34, June 1999.

11. S. E. Chen and L. Williams, “View Interpolation for Image Synthesis”, pp. 279-288, SIGGRAPH 93.

12. P. Debevec, C. Taylor, and J. Malik, “Modeling and Rendering Architecture from Photographs”, pp. 11-20, SIGGRAPH 96.

13. M. Levoy and P. Hanrahan, “Light Field Rendering”,pp. 31-42, SIGGRAPH 96.

14. A. Laurentini, “The Visual Hull Concept for Silhouette-based Image Understanding”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.16, No. 2, February. 1994.

15. J. Koenderink, “Solid Shape”, The MIT Press, 1990.

16. E. Boyer and M. Berger, “3D Surface Reconstruction Using Occluding Contours”, IJCV 22, pp. 219-233, 1997.

17. Naoya Ohta, “A Statistical Approach to Background Subtraction for Surveillance Systems”, Proceedings of ICCV, 2001.