Appearance-Based Virtual-View Generation for Fly Through in a Real Dynamic Scene

Shigeyuki Baba[1,2], Hideo Saito[1,3], Sundar Vedula[1], Kong Man Cheung[1], and Takeo Kanade[1]

[1] Carnegie Mellon University, Pittsburgh PA 15213, USA
[2] Sony Corporation, Tokyo 141-0001, Japan
[3] Department of Information and Computer Science, Keio University, Yokohama 223-8522, Japan

Abstract. We present appearance-based virtual view generation which allows viewers to fly through a real dynamic scene. The scene is captured by synchronized multiple cameras. Arbitrary views are generated by interpolating two original camera-view images near the given viewpoint. The quality of the generated synthetic view is determined by the precision, consistency and density of correspondences between the two images. All or most of previous work that uses interpolation extracts the correspondences from these two images. However, not only is it difficult to do so reliably (the task requires a good stereo algorithm), but also the two images alone sometimes do not have enough information, due to problems such as occlusion. Instead, we take advantage of the fact that we have many views, from which we can extract much more reliable and comprehensive 3D geometry of the scene as a 3D model. The dense and precise correspondences between the two images, to be used for interpolation, are derived from this constructed 3D model. Our method of 3D modeling from multiple images uses the Multiple Baseline Stereo method and Shape from Silhouette method.

1 Introduction

Recently, synthesizing virtual images from multiple real images have found many applications including virtual reality, tele-presence and stereoscopic displays. Some applications, such as off-line generation of appealing graphics, can use a manual procedure [2], but many others require automation. This requirement for automation is especially strong in dealing with dynamic scenes. This paper aims at a completely automated method for synthesizing virtual images.

Model Based Rendering is one technique for image synthesis. First, a 3D model is reconstructed from the multiple images. Then, the colors on the real images are used to form the texture of the 3D model. Using conventional rendering techniques, virtual images are generated from the color textured 3D model. Wheeler et al. [16] proposed a method for accurate 3D model reconstruction from multiple view range images and demonstrated that the generated 3D shape and reflectance model can synthesize high-quality virtual view images. Debevec et al. [2] created precise synthetic images of a static scene whose model is constructed

by an interactive 3D modeling system. Faugeras et al. [17] developed a system which can generate 3D models of a static environment semi-automatically. Our group [7] demonstrated automated creation of 4D (3D + time) models for time-varying scenes, together with texture mapping and rendering of new views. These methods have the advantage of handling the occlusion problem as they make use of the 3D models. However, texture mapping onto the constructed 3D model with errors may cause blur of synthesized virtual images.

Image Based Rendering method does not require any 3D models for synthesizing virtual images. Plenoptic methods that represents the radiance as a function of the position and the directions is one of the popular methods for Image Based Rendering [5, 9, 10]. The computation cost of these methods is less than that of Model Based Rendering. However, the creation of higher quality synthetic images requires a large number of original images. Another approach is generating synthetic images using correspondences between the original images such as View Interpolation [1] and View Morphing [13]. In those methods, correspondences between the original images must be specified for warping the original images to generate intermediate views. The correspondences are generally given manually [1, 13], by the use of optical-flow [3] or by the use of dense stereo matching [12, 15].

In this paper, we present a view interpolation approach which we call *Appearance-Based Virtual-View Generation*. First, a 3D model, which has enough geometrical information of a scene, is reconstructed from multiple images by using "Multiple Baseline Stereo" (MBS) [11] and "Shape from Silhouette" (SS) [4, 6]. Taking advantage of the fact that we have 3D models of the scene, geometrically accurate correspondences are derived from the 3D models. The precise and dense correspondences generate virtual views at arbitrary viewpoints without losing pixels even in occlusion regions. In the following sections, we describe the details of each process.

2 Three Dimensional Model Reconstruction

3D models are reconstructed from multiple images captured in a facility called "The 3D Room" [8] by using either MBS method or SS method depending on the complexity of an objects in any given scenes. The former method is used for complex objects and the latter method is used for simpler objects.

Before the execution of these methods, calibration of all cameras is required (we currently have 49 cameras in a room). We use Tsai's camera calibration algorithm [14]. Tsai's camera model has 11 parameters, consisting of five intrinsic parameters and six extrinsic parameters. To implement this algorithm, we built a calibration device using 64 LEDs on single plane. The 8x8 LEDs are placed at an interval of 300mm uniformly. Camera calibration images are taken at five different positions by changing the height of this device. Once we know the relationship between the image coordinates and the world coordinates by this measurement, all the camera parameters are computed with this algorithm.

In execution of MBS, some neighboring cameras are chosen for each of the 49 cameras. Depth images are generated for each camera and all of the depth images are merged into a single 3D model, which is represented by triangle meshes, by using volumetric merging at each time frame. The region of interest is specified during the execution of volumetric merging process in order to obtain the desired 3D objects in the dynamic scene.

As for Shape from Silhouette, foreground (silhouette) images are generated for each camera before the computation of 3D model. Background subtraction is performed for the input images from each of the 49 cameras and dilation and erosion processing is performed to improve the quality of foreground images. After generating foreground images of all cameras, all of the images are back-projected into 3D space. Each camera viewpoint and its foreground image define a bounding volume. The 3D model can be reconstructed from intersecting volumes of multiple bounding volumes defined by these foreground images.

3 Deriving Pairwise Correspondence from 3D Model

The 3D model reconstructed from multiple camera-view images is used to derive correspondences between any neighboring camera image pairs. Figure 1 shows how to derive the correspondences from the 3D model. View 1 and view 2 are the views of a pair of neighboring cameras. First, the intersection of the ray from the point a in view 1 with the surface of the 3D model is computed by using camera calibration data. Then, the intersecting point A on the surface of the 3D model is projected onto view 2 and the projected point a' is computed. That is, the point a' in view 2 is the corresponding point of the point a in view 1. If there are points whose pixel rays have no intersecting point on the surface of the 3D model like the points as shown figure 1, those points have no corresponding points. This procedure is performed for all the projected objects in view 1 and the correspondences from view 1 to view 2 are derived. We also need the correspondences from view 2 to view 1 to overcome occlusion problems.

After the derivation of the correspondences in the entire view, disparity vectors are defined for all of the corresponding points like the vector $\boldsymbol{d_a}$ and $\boldsymbol{d'_a}$ shown in figure 1. These vectors are used for estimating the pixel position in the virtual views.

4 Virtual View Generation

We extend the View Interpolation method, so that the correspondences, derived from a 3D model, can generate virtual views at arbitrary viewpoints without losing pixels even in occlusion regions.

4.1 Review of View Interpolation

Once the correspondences between two neighboring views are derived, synthesized views at arbitrary viewpoints between those views can be generated. We

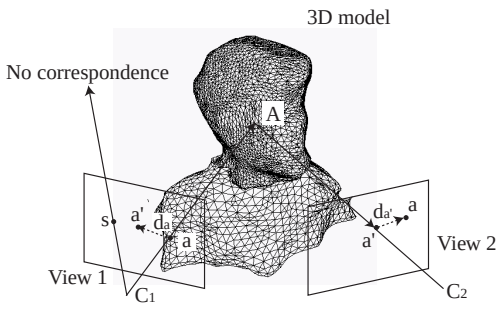

Fig. 1. The scheme for correspondence derivation between a pair of neighboring views, using a 3D model.

use an interpolation algorithm which is based on the related concepts of "View Interpolation" [1] and "View Morphing" [13] to generate the synthesized views. This interpolation algorithm involves the computation of the position and the color of the pixels using the correspondences between two images, described by the following equations.

$$\boldsymbol{P_i} = w_1 \boldsymbol{P} + w_2 \boldsymbol{P'} \ , \tag{1}$$

$$I_i(\boldsymbol{P_i}) = w_1 I(\boldsymbol{P}) + w_2 I'(\boldsymbol{P'}) \ . \tag{2}$$

where

$$w_1 + w_2 = 1$$

$\boldsymbol{P}$ and $\boldsymbol{P'}$ are the position of the corresponding points in the view 1 and the view 2, respectively. $I(\boldsymbol{P})$ and $I'(\boldsymbol{P'})$ are the color of the corresponding points in view 1 and view 2 as well. $\boldsymbol{P_i}$ is the interpolated position and $I_i(\boldsymbol{P_i})$ is the interpolated color and w_1 and w_2 ($w_1 + w_2 = 1$) are weighting factors.

4.2 Zooming

We have to account for the fact that focal lengths of various cameras may be different, when dealing with multiple cameras for capturing real views. In this situation, if two neighboring camera-views with different focal lengths are chosen, the object size in the virtual views change during the movement of viewpoints. To avoid this problem, it is necessary to add a zooming image feature to the view interpolation. We modify the view interpolation equation as follows;

$$\boldsymbol{P_i} = w_1 \left\{ (\boldsymbol{P} - \boldsymbol{C}) \frac{f_v}{f} + \boldsymbol{C} \right\} + w_2 \left\{ (\boldsymbol{P'} - \boldsymbol{C'}) \frac{f_v}{f'} + \boldsymbol{C'} \right\}, \tag{3}$$

$$I_i\left(\boldsymbol{P_i}\right) = w_1 I\left(\boldsymbol{P}\right) + w_2 I'\left(\boldsymbol{P'}\right) \ . \tag{4}$$

where

$$w_1 + w_2 = 1$$

$\boldsymbol{C}$ and $\boldsymbol{C'}$ are the optical centers in view 1 and view 2, respectively. f and f' are the focal lengths of camera 1 (view 1) and camera 2 (view 2). f_v is the focal length of the virtual camera. With this modification, the virtual camera can zoom in and out in accordance with the focal length f_v. This modification makes the view interpolation method more practical.

4.3 Viewport Transformation Using Calibration Data

Multiple cameras are usually installed facing towards the center of the object. However, it is difficult to adjust the center of the objects to the exact optical center of each camera-view, even for static objects. If there is an offset between the center of the objects and the optical center in the view, the objects in the virtual view may move out of the field of view during zooming as described in the previous section. To avoid this problem, we transfer the viewport so that the objects can be placed at the center of the virtual view.

Since the calibration data for each camera is computed, we can define the projection matrices using the intrinsic and extrinsic parameters for each camera. Then, if the center of the objects in the world coordinates is defined, it can be projected onto each view using those matrices. Comparing the position of this projected point and the optical center in the views, the transformation value for re-centering objects in views can be computed. Using these transformation values, the center of the objects can be shifted to the optical center in the virtual view.

4.4 Pseudo Correspondences for Handling Occlusion

It is not unusual that the camera views have occluded regions in the scene. For instance, if we have two cameras and a L shaped object in a scene, as shown in figure 2, a part of the surface may be in an occlusion region for those views. If we compute the correspondence point of q in view 1 by using the scheme described in the section 3, it is back-projected onto the occluded surface of the 3D model and there is no consistent corresponding point in the view 2. For such an occluded area, we can not generate interpolated views by the method described by equations (1) and (2).

We solve this problem by introducing the concept of *Pseudo Correspondences*. The pseudo correspondences can be derived from the 3D model and then applied to the view interpolation, described by the equations (1) and (2). In figure 2, the point q in view 1 has no corresponding point in view 2. The back-projected 3D point Q of the 2D point q on the occluded surface can be virtually re-projected onto the point q' in view 2, even though it can not be seen from view 2. We name such correspondences *"Pseudo Correspondences"* In addition, the point r is back-projected onto the point R on the surface of the 3D model and re-projected onto the point r' in view 2, the same point as q'. Hence, q' $(= r')$ has both the pseudo corresponding point q and the real corresponding point r in the view 1. Applying these pseudo correspondences to equation (1), the position of the pixels in the virtual view can be interpolated. However, the color of the pixel

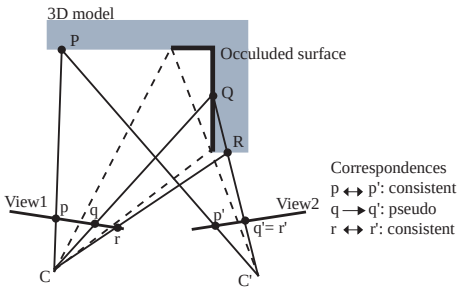

Fig. 2. Consistent and pseudo correspondences.

cannot be interpolated by equation (2) because the occluded surface cannot be seen from the view 2. In this case, the color should be simply chosen from the colors of the point in the neighboring two views.

To apply pseudo correspondences to view interpolation, we generate the two interpolated images using two directed correspondences, from view 1 to view 2 and from view 2 to view 1, separately. This means that two interpolated images are generated at one virtual view point. Then, the two interpolated images are blended into a single image. This implementation is described by the following equations:

$$I_w\left(u + w_1 d_u\left(u,v\right), v + w_1 d_v\left(u,v\right)\right) = I\left(u,v\right), \tag{5}$$

$$I'_w\left(u + w_2 d'_u\left(u,v\right), v + w_2 d'_v\left(u,v\right)\right) = I'\left(u,v\right) \quad .$$

where

$$d_u\left(u,v\right) = \left\{\left(u' - u'_c\right)\frac{f_v}{f'} + u'_c\right\} - \left\{\left(u - u_c\right)\frac{f_v}{f} + u_c\right\},$$

$$d_v\left(u,v\right) = \left\{\left(v' - v'_c\right)\frac{f_v}{f'} + v'_c\right\} - \left\{\left(v - v_c\right)\frac{f_v}{f} + v_c\right\},$$

$$d_u\left(u,v\right) = -d'_u\left(u,v\right), d_v\left(u,v\right) = -d'_v\left(u,v\right) \quad .$$

Both $I_w(u,v)$ and $I'_w(u,v)$ are the warped images generated by using the correspondences from view 1 to view 2 and from view 2 to view 1, respectively. $I(u,v)$ and $I'(u,v)$ are the original images at the two neighboring viewpoints. d and d' are the disparity vectors which are computed along with the derivation of the correspondences, described in the section 3. (u_c, v_c) and (u'_c, v'_c) are the optical centers in the view 1 and the view 2, respectively. These equations include zooming with focal length, f, f', f_v as described in the section 4.2.

After the generation of the two warped images, these are blended into a single interpolated image by using the following equation:

$$I_i\left(u,v\right) = \begin{cases} w_1 I\left(u,v\right) & \text{if } I\left(u,v\right) \neq 0 \text{ and } I'\left(u,v\right) = 0, \\ w_2 I'\left(u,v\right) & \text{if } I\left(u,v\right) = 0 \text{ and } I'\left(u,v\right) \neq 0, \\ w_1 I\left(u,v\right) + w_2 I'\left(u,v\right) & \text{otherwise} \quad . \end{cases} \tag{6}$$

The color of the warped pixel, generated by the pseudo correspondence, is simply chosen from either view 1 or view 2 like as in first two cases of this equation. Figure 3 shows the process of this view interpolation algorithm. View 1 and view 2 are the original views taken by two cameras. Two warped images are generated using the weighting factors, $w_1 = 0.6$ and $w_2 = 0.4$. Each warped image is generated by different correspondence data as described above. By blending these warped images, an interpolated view is generated. The circled areas in this figure show the occlusion regions in the views. Using conventional view interpolation, the color in these occluded area cannot be recovered. With pseudo correspondences, an interpolated view can be generated without losing pixels in these regions as shown in figure 3.

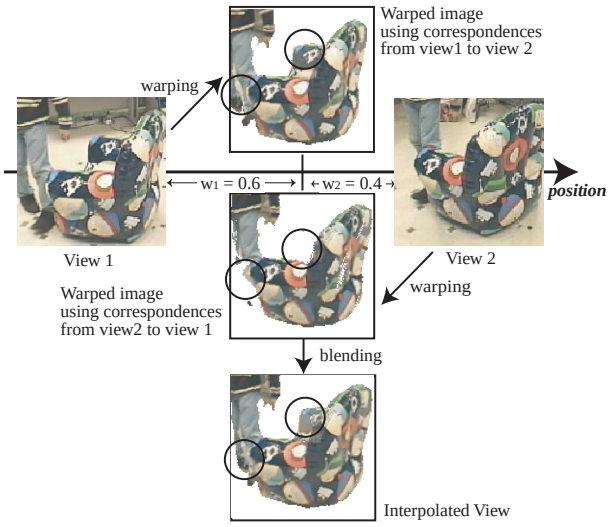

Fig. 3. View interpolation with pseudo correspondences.

5 Experimental Results

5.1 The 3D Room

In order to reconstruct 3D models of dynamic events efficiently and automatically, we have developed a facility called "3D Room" [8], in which dynamic events can be captured by synchronized cameras and be digitized as 3D representations with time frame. Forty nine cameras are installed at various points in the 3D room: 10 cameras on each of the four walls and 9 cameras on the ceiling. A PC cluster system (currently 17 PCs) can digitize all the NTSC video signals from the cameras simultaneously in real time as uncompressed and lossless color

images (YUV422) at full video rate (30 fps). Each PC is used for the digitization of the video signals as well as image processing.

5.2 Virtual View Generation with Various Weighting Factors

Figure 4 shows example results of the appearance-based virtual view interpolation. In this example, twelve interpolated images whose weighting factors are 0.0, 0.2, 0.8 and 1.0 are generated from the original images of the two cameras (cam #29 and cam #30) at three different time frames (0, 50 and 100). If the weighting factor is either 0.0 or 1.0, the quality of the interpolated images is the same as the original images of the camera #29 or the camera #30. As a result, when we view the scene from the same viewpoint as an original camera, we obtain the full quality image. This is one of the advantages of our view interpolation algorithm. In most of the model based rendering methods, the texture rendered onto the model surface is blurred by the error of the recovered 3D model, that results in blurred virtual view images even if the virtual view point is the same as the real camera position. Moreover, the occlusion area can be successfully interpolated in each virtual view because of the pseudo correspondences. We have developed a GUI-based viewer application for viewing virtual views, synthesized by using our methods. With this viewer, users can easily specify the virtual camera position by using a mouse and fly through a real dynamic scene.

5.3 Virtual View Generation from Camera-Views with Different Focal Length

Figure 5 shows another example of the appearance-based view interpolation. In this example, the original views are taken from two cameras with different focal length. In order to avoid changing the image size among the virtual view points, we use same focal length f_v of the virtual camera. Using the algorithm described in sections 4.2 and 4.3, the interpolated views are generated with same focal length and the objects in those views are successfully centered using the camera calibration data.

6 Conclusion

Our method, which we call the Appearance-Based Virtual-View Generation of dynamic events, uses a 3D model to derive accurate correspondences between the original views. We have defined *Pseudo Correspondences* in order to avoid the occlusion problems. Since our correspondences contain geometric information, virtual views are generated at arbitrary viewpoints without losing pixels even in occlusion regions. Virtual view generation based on Image Based Rendering can be implemented using simple and fast 2D image processing techniques. That is, once the correspondences are derived from the 3D model, processing time of the virtual view generation does not depend on complexity of the 3D objects like the other image based rendering methods. Zooming and centering features are

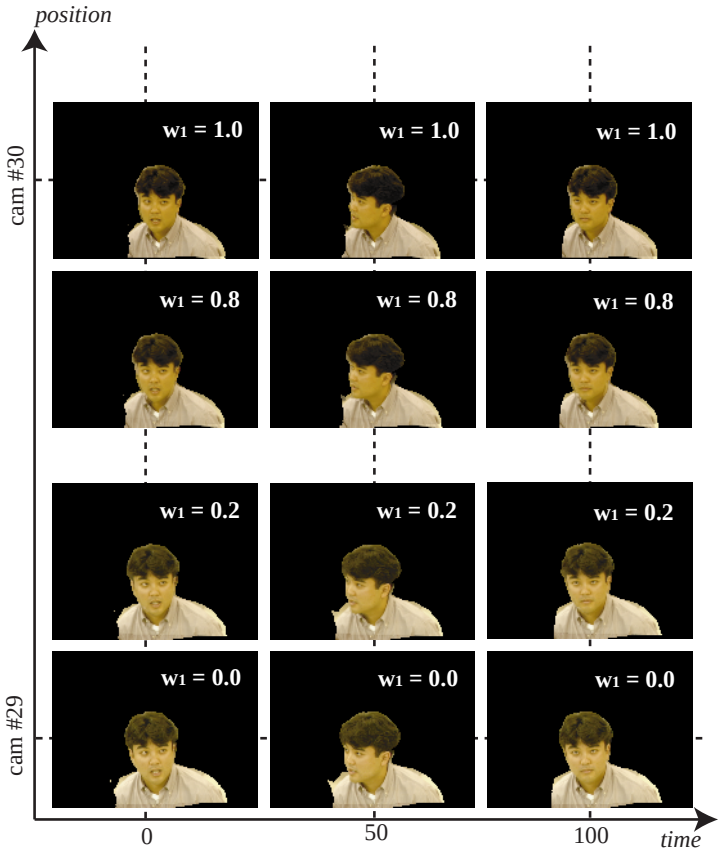

Fig. 4. Example results of Appearance based view generation.

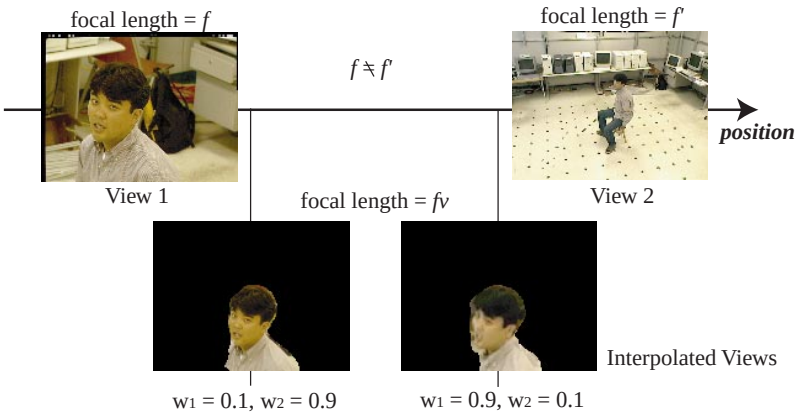

Fig. 5. Example results of Appearance based view generation, using two cameras with different focal length.

also implemented by using the transformation of the disparity vectors and the viewport. Thus the Appearance-Based Virtual-View Generation combines both accuracy and flexibility in the creation of virtual worlds from real views.

References

1. Chen, S., Williams, L.: View Interpolation for Image Synthesis. Proc. of SIGGRAPH'93. (1982) 279–288
2. Debevec, P., Taylor, C., Malik, J.: Modeling and Rendering Architecture from Photographs: A Hybrid Geometry and Image-Based Approach. Proc. of SIGGRAPH'96. (1996)
3. Avidan S., Shashua, A.: Novel View Synthesis by Cascading Trilinear Tensors. IEEE TVCG. Vol.4. No.4 (1998) 293–306
4. Potmesil, M.: Generating Octree Models of 3D Objects from Their Silhouettes in a Sequence of Images. Computer Vision, Graphics and Image Processing. **40** (1987) 277–283
5. Gortler, S.J., Grzeszczuk, R., Szeliski, R., Cohen, M.F.: The Lumigraph. Proc. of SIGGRAPH'96. (1996)
6. Chein, C.H., Aggarawal, J.K.: Identification of 3D Objects from Multiple Silhouettes using Quadtrees / Octrees. Computer Vision, Graphics and Image Processing. **36** (1986) 100–113
7. Kanade, T., Rander, P.W., Narayanan, P.J.: Virtualized Reality: Constructing Virtual Worlds from Real Scenes. IEEE Multimedia. Vol.4. No.1 (1997)
8. Kanade, T., Saito, H., Vedula, S.: The 3D Room: Digitizing Time-Varying 3D Events by Synchronized Multiple Video Streams. CMU-RI-TR-98-34 (1998)
9. Katayama, A., Tanaka, K., Oshino, T., Tamura, H.: A Viewpoint Dependent Stereoscopic Display Using Interpolation of Multi-Viewpoint Images. SPIE Proc. Vol.2409. Stereoscopic Displays and Virtual Reality Systems II (1995) 11–20
10. Levoy, M., Hanrahan, P.: Light Field Rendering. Proc. of SIGGRAPH'96 (1996)
11. Okutomi, M., Kanade, T.: A Multiple-Baseline Stereo. IEEE Trans. on PAMI. Vol.15. No.4 (1993) 353–363
12. Narayanan, P.J., Rander, P.W., Kanade, T.: Constructing Virtual Worlds Using Dense Stereo. Proc. ICCV'98 (1998)
13. Seitz, S.M., Dyler, C.R.: View Morphing. Proc. of SIGGRAPH'96 (1996) 21–30
14. Tsai, R.: A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-The-Shelf TV Cameras and Lenses. IEEE J.of Robotics and Automation RA-3. 4. (1987) 323–344
15. Vedula, S., Rander, P.W., Saito, H., Kanade, T.: Modeling, Combining and Rendering Dynamic Real-World Events from Image Sequences. Proc. 4th Conf. Virtual Systems and Multimedia. Vol.1 (1998) 326–332
16. Wheeler, M.D., Sato,Y., Ikeuchi, K.: Consensus Surfaces for Modeling 3D Objects from Multiple Range Images. DARPA Image Understanding Workshop (1997)
17. Faugeras, O., Laveau, S., Robert, L., Csurka, G., Zeller, C.: 3-D Reconstruction of Urban Scenes from Sequences of Images. INRIA Technical Report. No.2572 (1995)