PHOTOREALISTIC OBJECT RECONSTRUCTION USING VOXEL COLORING AND ADJUSTED IMAGE ORIENTATIONS

Yasemin Kuzu, Olaf Sinram
Photogrammetry and Cartography
Technical University of Berlin
Str. des 17 Juni 135, EB 9
D-10623 Berlin, Germany
{kuzu, sinram}@fpk.tu-berlin.de

ABSTRACT

In this paper we present a voxel-based 3D reconstruction technique to compute photo realistic volume models from multiple color images. Visibility information is the key requirement for photorealistic reconstructions. In order to get the full visibility information item buffers are created for each image. In the item buffers each pixel is assigned the ID of the closest voxel it corresponds to. All scene points are projected into every image and if the ID stored in the pixel is the same as the voxel's ID, they are checked for photo consistency. This way only visible (non-occluded) voxels are considered. Voxels are labelled either opaque and assigned a color value or transparent and carved away until no non-photo consistent voxel exists. Due to the missing of control points on the object, the images undergo a process of relative orientation in a free network using manually measured tie points. With these image orientations and a previously calibrated camera, the scene points are projected in a high accurate way, minimizing projection errors. This way objects are modelled where the application of control points is either impossible (e.g. in microscopic images) or uneconomical. Experimental results show that the combination of photogrammetric methods and shape from photo consistency techniques yield more accurate results since camera errors are taken into consideration and images are assigned individual orientation data.

INTRODUCTION

Computer vision concerns with developing automatic extraction of various types of information from imagery. One of its purposes is creation of shape and surface properties of arbitrarily shaped 3D solid objects from its 2D images. Furthermore, extracting 3D information from photographs for obtaining precise measurements has a long history in photogrammetry. There is a growing need for 3D model reconstruction in many fields such as virtual reality, CAD/CAM, robotics and entertainment. 3D object acquisition can be performed by range sensors such as laser scanners (Curless and Levoy, 1996) or by image sequences from a monoscopic camera, which is a simple and a low cost alternative therefore attractive for all these applications.

Conventional computer graphics typically create images from a model using its geometric descriptions, which is a labor-intensive process for complex scenes and the result may not look realistic. Image based rendering however, has attracted significant interests because images provide tremendous details, yet can be acquired fast, cheap and easily even with inexpensive amateur cameras.

Volumetric representations use voxels to model the scene, which consume large amounts of memory, for example 256^3 bytes (16.77 mbytes) for a rather small cube (256 units in each direction). With the rapid advances in hardware however, this becomes less of a problem and volumetric representations become more attractive.

The earliest attempts of volumetric scene representations from photographs are based on volume intersection methods. These methods are often referred to as shape from silhouette algorithms. The intersection of silhouette cones from multiple images defines an estimate geometry of the object called the visual hull. This technique gives a good approximation of the model. However, the concavities on an object cannot be recovered since the viewing region doesn't completely surround the object.

The recent attempts are based on voxel coloring algorithms. Photorealistic modelling is a difficult and important problem in computer graphics. In many new applications like automatic 3D model reconstruction, telepresence, virtual walkthroughs there is a need for developing photorealistic models of real environments from images that look and move realistically. In this paper we consider the problem of computing photorealistic volume models of real environments from a set of color images of the scene.

Voxel coloring algorithms reconstruct a 3D scene that is consistent with its input images. Color consistency is introduced by (Seitz, 1997) to distinguish surface points from the other points in the scene making use of the

fact that surface points in a scene project into consistent (similar) colors in the input images. Voxels that are consistent with the images are assigned their corresponding color value and inconsistent voxels are removed from the voxel space. The algorithm continues until no non-photo consistent voxels exist and the final set of voxels contain sufficient color and texture information to accurately reproduce images of the scene.

In order to reconstruct the real world, the camera setup including the interior and exterior orientations has to be remodelled. In the following work we describe the calibration of the camera and the subsequent orientation of the image bundle.

RELATED WORK

Numerous researchers have dealt with the shape from silhouette technique to convert visible contours into a visual hull, i.e. (Szeliski, 1991), (Niem, 1994), (Kutulakos and Seitz, 1998) and (Vedula et al., 1998).

Voxel Coloring (Seitz and Dyer, 1997), Space Carving (Kutulakos and Seitz, 1998) and Generalized Voxel Coloring (Culbertson et al, 1999) have been recent successful solutions to the 3D scene reconstruction problem. Voxel coloring algorithms determine the voxels that are on the surface of the object therefore they are photo consistent with the input images. The consistency of a set of pixels can be defined as their standard deviation. The voxel is considered to be on the surface if the derived standard deviation is less than a given threshold. Seitz and Dyer's voxel coloring algorithm restricts the position of the cameras in order to simplify visibility information. Their algorithm works coloring the voxels in a special order similar to Collins' Space-Sweep approach (Collins, 1996), which performs an analogous scene traversal.

Kutakulos and Seitz's space carving theory removes the constraint of camera placement. However space carving algorithm does not use the full visibility of the scene. Both voxel coloring and space carving scans voxels for color consistency by evaluating a plane of voxels at a time.

Culbertson, Malzbender and Slabaugh introduced an efficient voxel coloring algorithm called Generalized Voxel Coloring (GVC) that computes visibility exactly and yield a color consisting model. Unlike Voxel Coloring, GVC removes the constraint on camera positions. On the other hand it uses all images to check voxel consistency unlike Space Carving, which uses only a subset of the images. There are two variants of the algorithm called GVC and GVC-LDI. When a voxel is carved it changes the visibility of the other voxels. GVC uses item buffers and GVC-LDI uses layered depth images to keep track of the scene visibility. GVC-LDI exactly determines which voxels have their visibility changed when a voxel is carved while GVC revaluates all voxels in the current model. Therefore GVC-LDI algorithm requires fewer color consistency evaluations however GVC uses considerably less memory than GVC-LDI.

We will refer to generalized voxel coloring in this work and present some high accurate results with real objects by combination of photogrammetric and shape from photo consistency techniques.

SYSTEM CONFIGURATION

Experimental Setup

The experimental system configuration uses a single stationary video CCD camera and a turntable as a controlled motion platform on which the object is placed. In Fig.1 the virtual camera positions are shown for the calibration (left) and the object reconstruction of the sample (right). Since the camera is fixed and the coordinate system is tied to the turntable and therefore rotating itself, we show virtual camera positions.

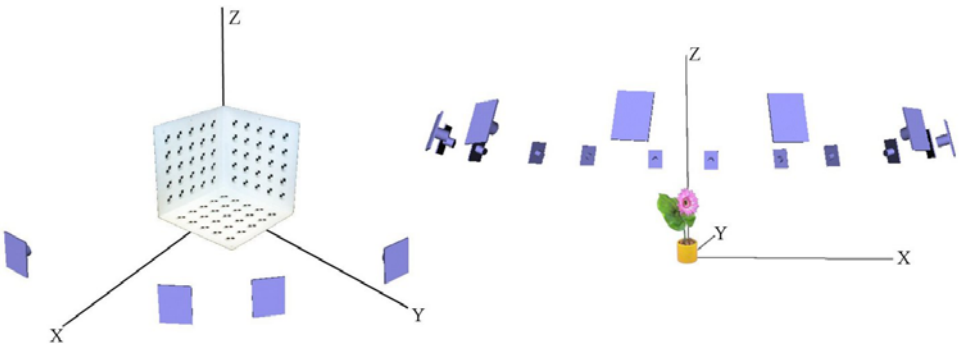

Figure 1: Virtual camera positions according to the turntable-fixed coordinate system.

Camera Calibration

Before starting the image acquisition, the system must be calibrated. The interior and exterior parameters of the camera must be known, i.e. the position and orientation of the camera and the principal distance. We perform a system calibration by using a special calibration object (cp. Fig.1). It provides a good coverage of the object with three square faces containing 25 control points on each side. The geometric description of the configuration using central projection without a lens distortion model is shown in Fig. 2:

O : Projection center with the coordinates X_0, Y_0, Z_0
H : Principal Point with the coordinates x_0, y_0
M : Image Center
c : Principal distance (camera constant)
P : Real - World Point with the coordinates X, Y, Z
P': Image Point with the coordinates x, y, z

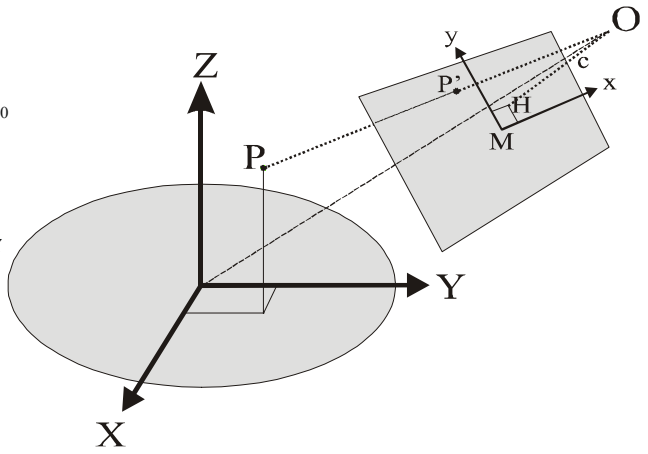

Figure 2: Mathematical camera model for central projection.

Image Orientation

In order to project voxels to images the interior and exterior orientation must be known. The interior orientation, meaning the focal length and the principal point of the camera, is calculated in a separate camera calibration process as seen in Fig. 1 (left). For the exterior orientation parameters (the location and rotation of the projection center), we are able to get very good estimation values, due to the scene setup, which allows the turntable to be rotated in controlled steps. However, since we use a non-calibrated turntable, the orientation parameters turned out to be quite noisy. Therefore the projections of the voxels would not yield the true pixel, instead the projection spread around the true position.

Theoretically the image orientations should only be dependent on the rotation angle of the turntable. Two rotations (ν, κ), Z_0 and the distance of the projection center to the coordinate system's origin should be constant. The third rotation (α) is a direct function of the turntable position. But since the setup is not high accurate, for example a small eccentricity of the coordinate system's origin, would result in changing parameters.

The introduction of independent, individual orientation parameters for the images, removes the limitations concerning the scene setup. The images can be taken from arbitrary positions.

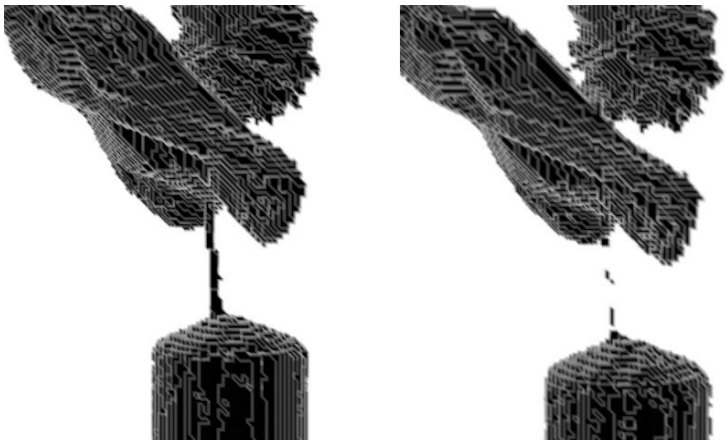

Figure 3: Volumetric intersection with adjusted (left) and estimated (right) orientation parameters.

Fig. 3 shows the consequences of inaccurate parameters to the volume intersection. On the left image adjusted parameters were used, minimizing the projection error. On the right image we see the result when using the estimated orientation parameters. Due to the errors in projections some voxels might be falsely considered as background, which will delete too many voxels. In Fig. 3 it is clearly visible that the stem disappeared almost completely. As seen on the left image however, it remains when projected with adjusted parameters.

The orientation was done in two steps. First the interior orientation had to be calculated by a camera calibration as seen in Fig. 1 (left). This process also yields one camera position when replacing the calibration object with the actual object. With this first image and a number of tie points the camera setup can be reconstructed except for the scale. This can be eliminated using one single control point. As an alternative, a distance between two arbitrary points can be taken as well, if no control points exist at all.

The whole bundle was now adjusted with the following parameters:

12 known parameters (c, x_0, y_0, α, ν, κ, X_0, Y_0, Z_0, X_P, Y_P, Z_P for camera, one image and one control point)
267 unknowns (15·6 for the images, 59·3 for object points)
638 observations (319 observed tie points)

It yielded the image orientations for all 16 images, see Fig. 1 (right), as well as object coordinates for the observed tie points. Those object points were not the goal but just a byproduct, because the very object will be reconstructed by the techniques described in the following chapters. After the adjustment the image orientations were calculated with an accuracy of 5-7 mm in position and 0.3 and 0.5 gon in rotation. However, the introduction of control points can only improve the results.

AUTOMATIC IMAGE SEGMENTATION

Background Estimation

In our experiment, the turntable is rotated and the video images are captured and the contour of the real object is extracted. Therefore, a monochromatic background was used to distinguish the object from the environment. For a small frame (see dotted line in Fig.4) the background color is estimated using a histogram of the hue values.

Figure 4: Background estimation using an IHS color space histogram (left) and the resulting silhouette extraction (right).

Image Segmentation

For every color pixel (R,G,B) a transformation into the IHS color space is computed:

$$I = \frac{R+G+B}{3} \qquad S = 1 - \frac{\min(R,G,B)}{I}, \qquad I > 0.05$$

$$H = \arccos\left(\frac{(R-G)+(R-B)}{\sqrt{(R-G)^2+(R-B).(G-B)}}\right), \quad B < G$$

The automatic segmentation algorithm compares all the hue data with the estimated background information using a simple threshold and a minimum saturation value. Finally, a morphological operation ('opening') eliminates isolated pixel.

The automatically derived results where then checked with the academic software HsbVis (HSB-Visualization) that allows interactive segmentation and color channel splitting and merging on a graphical user interface.

PHOTOREALISTIC OBJECT RECONSTRUCTION

We begin the algorithm by initializing a volume that encloses the 3D object to be reconstructed (Kuzu, Rodehorst, 2001). We used a rough bounding volume found by the volume intersection method. Although this volume is a good approximation of the model, it does not fully cover the concavities. We kept this volume large enough to bound the scene. According to this volume a list of all boundary voxels that completely separate empty regions of space from opaque regions are created and defined as surface voxel list (SVL). The SVL should be updated during the reconstruction iterations to make sure that only a minimum number of voxels are processed.

Color Consistency

Voxel coloring algorithms use the fact that only surface voxels of the scene project into equal colors in the input images. In Fig. 5 the surface voxel A is projected into the same colors in the images while the voxel B that is not on the surface is projected into different colors. In the presence of noise and quantization effects the consistency of a set of pixels can be defined as their standard deviation. The voxel is considered to be on the surface if it is less than a sensible threshold, which varies according to the object, the sensor and lighting conditions. The voxels that exceed this threshold are considered non-surface voxels and discarded or carved.

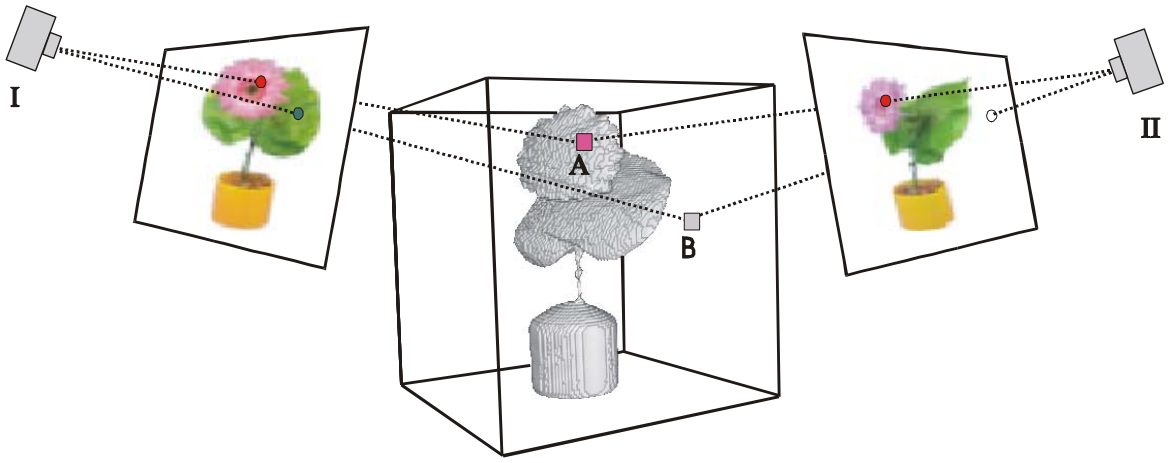

Figure 5: Color consistency is used to distinguish surface points from points not on the surface. Voxel A is seen red on both cameras while voxel B is seen green on camera I and transparent (background) on camera II.

Visibility computation

An item buffer is constructed for each image in order to obtain the full visibility information. Item buffers are used to store the visible voxels in the scene from a given viewpoint. In the item buffer, each pixel has the ID of the nearest voxel it projects into. In Fig. 6 (left) both voxel A and voxel B project into the same pixel. Since voxel A occludes voxel B, in the item buffer the ID of voxel A is recorded in the corresponding pixel.

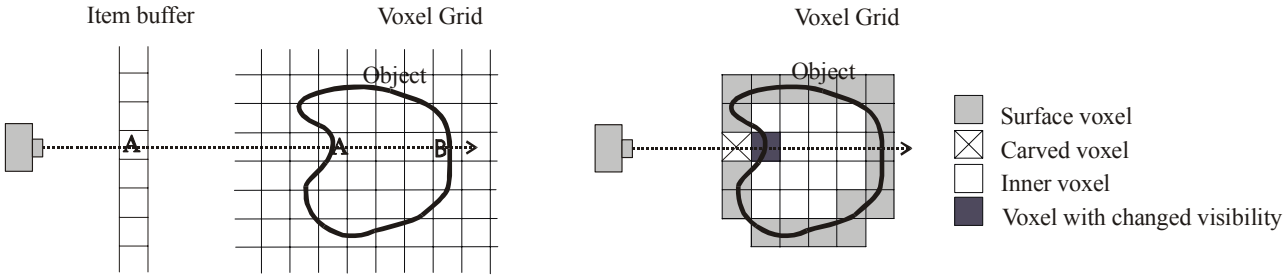

Figure 6: An item buffer records the ID of the closest surface voxel visible from each pixel in an image (left). When a voxel is carved, the adjacent inner voxels have their visibility changed (right).

The projection of voxels into the image plane covers more than one pixel due to its cubic shape. Depending on the scale of the whole scene a single voxel may project onto a footprint of pixels (for large images) or several voxels merge into one pixel (for small images). If the exact projection of voxels is not found, there might be gaps in the image's item buffer. Through these holes far away voxels may shine through, storing their ID in the item buffer, although they should not be visible. This condition might lead to false visibility information and failure of color consistency check. Therefore the item buffer should be dense, meaning that each pixel should contain the nearest voxel's ID.

Hence, instead of taking the footprint of voxels we traced the pixels back into the scene to find the first opaque voxel along the line of sight in the cube. To find the minimum and maximum factors in the loop boundaries, we derived the lambda-factor (see equation below) for each corner of the entire voxel cube. The loop starts with greater step size, which decreases as soon as the cube is entered therefore the process is accelerated. The loop ends when either a surface voxel is encountered, or the ray exits the cube, not encountering any opaque voxel at all. This way each pixel, which does not have a background value, is assigned the ID of a voxel, without leaving gaps in the image item buffer. This approach would correspond to the indirect image resampling methods, which does a geometric transformation indirectly from the destination back to the source.

The algorithm uses the image coordinates and performs a projection into voxel coordinates using the following equations:

$$\lambda \begin{pmatrix} x_i - x_0 \\ y_i - y_0 \\ -c \end{pmatrix} = R \begin{pmatrix} X - X_0 \\ Y - Y_0 \\ Z - Z_0 \end{pmatrix} \qquad \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = R^{-1} \lambda \begin{pmatrix} x_i - x_0 \\ y_i - y_0 \\ -c \end{pmatrix} + \begin{pmatrix} X_0 \\ Y_0 \\ Z_0 \end{pmatrix}$$

From the equations above we can derive the formulas:

$$\begin{aligned} X &= \lambda r_{11}(x_i - x_0) + \lambda r_{12}(y_i - y_0) - \lambda r_{13} c + X_0 \\ Y &= \lambda r_{21}(x_i - x_0) + \lambda r_{22}(y_i - y_0) - \lambda r_{23} c + Y_0 \\ Z &= \lambda r_{31}(x_i - x_0) + \lambda r_{32}(y_i - y_0) - \lambda r_{33} c + Z_0 \end{aligned}$$

Where the parameters r_{ik} are the elements of the inverse rotation matrix and λ is the scale factor. This is the factor by which the distance of the image point to the projection centre must be multiplied to obtain the distance from the object point to the projection centre.

Voxel coloring

Once the item buffers are calculated for each image, each voxel on the surface voxel list is projected into each image. This is the inverse transformation compared to the creation of the ID buffers. Following basic equations are used for this direct projection:

$$x_i = x_0 - c\frac{R_{11}(X - X_0) + R_{21}(Y - Y_0) - R_{31}(Z - Z_0)}{R_{13}(X - X_0) + R_{23}(Y - Y_0) - R_{33}(Z - Z_0)}$$
$$y_i = y_0 - c\frac{R_{12}(X - X_0) + R_{22}(Y - Y_0) - R_{32}(Z - Z_0)}{R_{13}(X - X_0) + R_{23}(Y - Y_0) - R_{33}(Z - Z_0)}$$

The ID of the projected voxel is compared to the ID stored in the image's ID buffer. If they are equal the pixel is added to the visibility list and the colors are recorded. Then all the visible pixels are checked for color consistency. If the standard deviation for these pixels exceeds a predefined threshold it is removed or carved from the volume. Otherwise it is given the average color value of the visible pixels.

When a voxel is carved the adjacent opaque voxels become surface voxels as seen in Fig. 6 (right). The previously derived SVL becomes invalid and has to be re-evaluated in the next iteration. The algorithm stops when all the inconsistent voxels are removed. The remaining surface voxels are photo consistent with the input images and form the photorealistic 3D model of the scene. Fig. 7 shows the pseudo code for the algorithm.

The final result can be seen in Fig. 8. It has been computed as a 256^3 voxel cube, using 16 images, with a resolution of 1020 by 1360 pixels, each.

```
loop
{
   initialise item buffers
   initialise SVL
   for every voxel V on SVL
   {
         find the set S of image pixels from which V is visible
         if V is consistent
         {
               color V
         }
         else
         {
               carve V
         }
   }
   if all voxels are consistent
         quit
}
```

Figure 7: Pseudo-code for the algorithm.

Figure 8: Reconstruction of a flower by voxel coloring using 16 images.

SUMMARY AND FUTURE WORK

We presented an approach to construct photorealistic 3-D models of real-world objects. The system requirements are simple therefore the method is attractive for many applications. A relative image orientation was followed by volume intersection and refined by a voxel coloring algorithm.

In order to achieve good results for the image orientations, spatially well distributed tie points had to be measured carefully, since the whole system is only based upon these measurements, besides the calibration data.

The visibility computation is slow, but very accurate. We would like to investigate the possibilities of speed optimisation and a more reliable way for color consistency check under different environmental conditions.

Future work will concentrate on different tasks. On the one hand the current algorithms are checked for further automation, for example the automatic measurement of tie points and the elimination of errors in the adjustment process. Furthermore we would like to investigate the application of the algorithms to other objects under different circumstances and varying setups.

A current project at the Department for Photogrammetry and Cartography is dealing with the reconstruction of biological objects with images taken by scanning-electron-microscopes[*]. We hope to transfer the experiences made here into other projects at hand. For the reconstruction of SEM images a tilting sample stage has been developed, corresponding to the turntable here. By the application of relative orientation, taking different laws of projection into consideration, the evaluation of microscopic 3D data should become more accurate. It should also set us free of limitations in the tilting setup. Also a microscopic calibration object is available. The image segmentation will not be applicable due to the incapability of an object-background-differentiation, but the derivation of spatial data may benefit from this work, resulting in either voxel or polygonal data.

Acknowledgements

The authors would like to thank V. Rodehorst for his contributions, J. Moré and A. Wiedemann for their helpful assistance to the camera calibration and bundle block adjustment task.

REFERENCES

Collins, R. T. (1996). A space-sweep approach to true multi-image matching. In *Proc. Computer Vision and Pattern Recognition Conf.*, pages 358–363.

Culbertson, W. B. Malzbender, T. Slabaugh, G. (1999). Generalized Voxel Coloring. *Proc. of the Vision Algorithms, Workshop of ICCV*, pp. 67 – 74.

Curless, B. and Levoy, M. (1996). A Volumetric Method for Building Complex Models from Range Images. *SIGGRAPH 96*, 303-312.

Kutulakos, N. and Seitz, M. (1998). A Theory of Shape by Space Carving. *University of Rochester CS Technical Report 692*.

Kraus, K. (1997). *Photogrammetry. Dümmlers Verlag*, pp. 15-19.

Kutulakos, K. N. and Seitz, S. M. (1998). What Do N Photographs Tell Us about 3D Shape? *TR680, Computer Science Dept. U. Rochester*.

Kuzu, Y. Rodehorst, V. (2001). Volumetric Modeling using Shape from Silhouette. *Fourth Turkish-German Joint Geodetic Days*., pp. 469-476.

Laurentini, A. (1995). How far 3D Shapes Can Be Understood from 2D Silhouettes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.17, No.2.

Matusik, W. Buehler, C. Raskar, R. Gortler, S. J. and McMillan, L. (2000). Image-Based Visual Hulls. *SIGGRAPH 2000 , Computer Graphics Proceedings, Annual Conference Series*, pp. 369-374.

Niem, W. (1994). Robust and Fast Modeling of 3D Natural Objects from Multiple Views. *Proceedings "Image and Video Processing II"*, Vol. 2182, pp. 388-397.

Seitz, M. Dyer, R. (1997). Photorealistic Scene Reconstruction by Voxel Coloring. *Proceeding of Computer Vision and Pattern Recognition Conference*, pp. 1067-1073.

Slabaugh, G. Culbertson, B. Malzbender, T. and Schafer, R. (2001). A survey of methods for volumetric scene reconstruction from photographs. In *International Workshop on Volume Graphics*, Stony Brook, New York.

Szeliski, R. (1997). From images to models (and beyond): a personal retrospective. *Vision Interface '97, Kelowna, British Columbia, Canadian Image Processing and Pattern Recognition Society*, pp. 126-137.

Szeliski, R. (1991). Shape from rotation. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'91),* pp. 625-630.

Vedula, S. Rander, P. Saito, H. and Kanade, T. (1998). Modeling, Combining, and Rendering Dynamic Real-World Events From Image Sequences. *Proc. 4th Conference on Virtual Systems and Multimedia (VSMM98)*, pp. 326-332.

---

[*] Project at the Department of Photogrammetry and Cartography in cooperation with the Heinrich-Pette-Institute Hamburg, supported by Deutsche Forschungsgemeinschaft