VOLUMETRIC RECONSTRUCTION OF CULTURAL HERITAGE ARTIFACTS

Y. Kuzu, O. 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

Commission V, WG V/2

KEY WORDS: Calibration, Close-Range, Correlation, Cultural Heritage, Photo-realism, Reconstruction, Visualization, Volume Model.

ABSTRACT:

In this paper we present a voxel-based object reconstruction technique to compute photo realistic volume models of cultural heritage artifacts from multiple color images. We use a CCD camera to acquire images of historical artifacts from several viewpoints. After calibrating the camera, the orientation parameters of the acquired images are calculated with a bundle block adjustment. Since it is impossible to mark control points on historical artifacts, natural texture was used. Our algorithm begins with initializing a volume that encloses the 3D artifact to be reconstructed. A first approximation of the model is acquired by shape from silhouette which gives the objects visual hull. Unfortunately, this method does not recover concavities on the object. In order to refine the representation, several tools will be described. An important factor is the visibility information of a voxel in a specific image. It can be either on the backside of the object, or occluded by other voxels in between. We present a fast method to recover this information with a line tracing algorithm. Furthermore, a quality measure of the visibility is introduced, by using the surface normal vector of a voxel in combination with the viewing direction of the image. These tools will help to generate the upgraded model more accurately.

KURZFASSUNG:

In dieser Arbeit stellen wir eine Technik zur Voxel-basierten Objektrekonstruktion vor, um fotorealistische Volumenmodelle von Objekten aus mehreren Farbbildern zu erstellen. Wir verwenden eine CCD-Kamera um von verschiedenen Standpunkten Bilder dieser Objekte aufzunehmen. Nach der Kalibrierung der Kamera werden die Orientierungsparameter der Bilder mit einer Bündelblockausgleichung berechnet. Da es nicht möglich ist, Passpunkte auf dem Objekt anzubringen, wird natürliche Textur verwendet. Am Anfang unseres Algorithmus wird ein Volumen initialisiert, welches das dreidimensionale Artefakt vollständig umschließt. Eine erste Näherung des Modells wird mittels „Shape from Silhouette“ berechnet, welches die umgebende Hülle bildet. Leider kann diese Methode keine konkaven Regionen des Objektes erfassen. Verschiedene Methoden werden beschrieben, die diese Näherung verfeinern. Ein wichtiger Faktor ist die Sichtbarkeit eines Voxels in einem bestimmten Bild. Ein Voxel kann sich auf der Rückseite eines Objekts befinden, oder durch andere Voxel innerhalb der Sichtrichtung verdeckt sein. Wir stellen eine Methode vor, die mittels eines Linienverfolgungs-Algorithmus sehr schnelle Information liefert. Darüber hinaus verwenden wir die Oberflächennormale des Voxels als ein Qualitätsmerkmal für die Sichtbarkeit bezüglich der Blickrichtung eines bestimmten Bildes. Diese Werkzeuge bilden die Grundlage, das Modell zu verfeinern.

1. INTRODUCTION

Reconstruction of the shape of 3D objects from a series of images is a challenging problem both in the disciplines photogrammetry and computer vision.
When generating 3D models with traditional CAD techniques, the details of 3D objects should be entered manually using graphical interfaces. Furthermore, with traditional CAD techniques objects are modeled with polygon patches and traditional material description which does not give high degree of realism. Since it is a labor intensive and complex process, the efforts concentrated in automatic modeling of 3D objects either with active methods by scanning the objects or with passive methods by using their images taken by a CCD camera. In this paper, we describe an efficient image-based approach to compute volume models of cultural heritage artifacts from their color images which is a low cost therefore an attractive method.

Cultural heritage preservation is one of the application areas of 3D model reconstruction. A full three-dimensional reconstruction serves as a permanent record of the heritage artifacts in their original position. Such a reconstruction can be used to detect the changes for conservation purposes. The models may also serve as manufacturing blueprint for machine production of replicas for exhibitions. If the object is an historical device for example the visitors might like to see it in action and perform their experiments. High precise reconstruction of the artifacts or replica can be made accessible to scholars and visitors.
Another application area is virtual museums where 3-D reconstruction, modeling and visualization of cultural heritage artifacts can be done volumetrically. A virtual museum is a computer generated environment where the artifacts and information resources of the museum can be viewed locally or on the internet. The users can view these artifacts from different

angles and distances. Besides, these representations of historical artifacts are enriched by online explanations, animations, music, video and narrations. This allows virtual museum visitors use this computer generated environment interactively for cultural research or educational purposes. In virtual museums there might be large numbers of dynamic virtual objects; therefore virtual objects should be modeled effectively. In order to increase realism of the virtual world, objects can be modeled from their images.

In this paper we use voxel-based methods to recover the shape of the cultural heritage artifacts. Voxel methods consume large amounts of memory, for example 512^3 bytes (128 mbytes) for a medium size cube (512 units in each direction). Since there are rapid advances in hardware however, this problem is becoming less important and volumetric representations is becoming more attractive.

The model of the 3D object can be easily acquired by shape from silhouettes methods in which the shape of the objects is recovered by intersecting the volumes. The intersection of silhouette cones from multiple images gives a good estimation of the true model. This approximate model is called the visual hull (Laurentini, 1995), (Matusik et al, 2000). Shape from silhouette methods is fast and robust; however the concavities and the critical areas on an object cannot be recovered with this method because the viewing region doesn't completely surround the object. The later work to recover the shape of the objects from multiple images is concentrated on voxel coloring algorithms (Seitz and Dyer, 1997), (Culbertson et al, 1999), (Kuzu and Sinram, 2002). These algorithms use color consistency to distinguish surface points from the other points in the scene. They use the fact that surface points in a scene project into consistent (similar) colors in the input images.

In this paper we describe several tools to refine the object's visual hull.

The organization of the chapters is as follows: In chapter 2 the image acquisition setup is described. The image orientation process is also explained. The reconstruction of the model using shape from silhouette technique will be described in chapter 3, where the refinement tools are introduced as well. Also there, an effective method to recover visibility information will be introduced. Furthermore, this information will be enhanced with a quality measure, by using the surface normal vector of a voxel in combination with the viewing direction of the image. In chapter 4 the refinement algorithm is explained.

2. IMAGE ACQUISITION SETUP

The system requirements of our experiment is simple, we use a CCD video-camera in order to acquire still images of the artifacts. Moreover, we use a calibration object to compute the interior orientation parameters of the camera. We place the object in front of a blue background. Image segmentation is a requirement to recover visual hull of the object. In order to segment object pixels from background pixels we place the object in front of a homogeneous blue background. We capture multiple views of the object resulting in a circular camera setup.

2.1 Camera Calibration and Determination of Control Points

Before acquiring the images, the camera should be calibrated. In our experiment we use a standard CCD video camera with auto focus. The focus of the camera can be fixed but we cannot tell whether it is unchanged since the last use. Hence, we calibrated the sensor using several images with a special calibration object which provides a good coverage of the objects having three perpendicular square planes and 25 control points on each side.

In a second session, the object is placed inside the calibration frame in order to define some natural control points accurately, as shown in figure 1.

A bundle block adjustment including all the images delivered not only the interior camera parameters, but also the coordinates of new points on the vase, which will serve as control points in the space carving processes.

During the subsequent image acquisitions, the focus remained fixed.

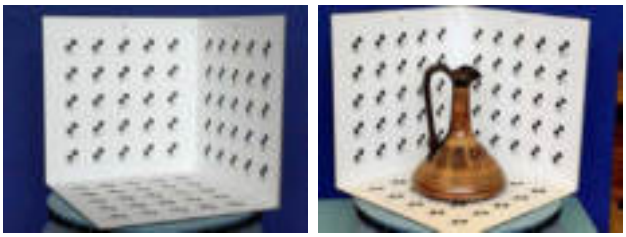

Figure 1: Camera calibration and control point determination

2.2 Image Orientation

In order to model the objects accurately, the images should be oriented. As mentioned above, we determined some natural control points on the objects surface, since we should not put markings on an historical artifact. They were defined in an arbitrarily chosen coordinate system, since there is no need to have the coordinates in a specific higher-level coordinate system.

The images were adjusted in a bundle block adjustment process. We used enough tie points in all images in the circular camera setup to perform a bundle block adjustment, covering all images. Figure 2 shows an OpenGL visualization of the situation. We achieved very accurate results for the image orientations, using the previously calibrated camera.

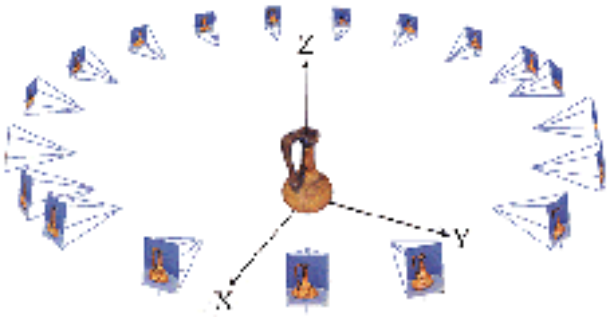

Figure 2: The virtual camera setup.

3. VOXEL-BASED ALGORITHMS

3.1 Shape from Silhouette

Shape from silhouette is a well-known approach for recovering the shape of the objects from their contours. This approach is popular in computer vision and in computer graphics due to its fast computation and robustness.

As a precondition of volume intersection algorithms, the contour of the real object must be extracted from input images. In this experiment a monochromatic blue background was used

to distinguish the object from the environment. The pixels position in the IHS-colorspace is examined in order to decide if it represents background or object. Since the blue background is sufficiently homogeneous, we can easily define a hue domain which is considered background. We performed the image segmentation using the academic software HsbVis (HSB-Visualization) that allows interactive segmentation and color channel splitting and merging on a graphical user interface.

Shape from silhouette algorithms start with an opaque volume that encloses the entire scene. This volume is discretized into voxels. If we know the calibration information and image orientation data, we can construct a bounding pyramid for each silhouette image. Each point in silhouette defines a line in object space that intersects the object at some depth. The combination of all these rays for all foreground silhouette points defines a cone in which the objects should lie. In order to compute the silhouette cone, we projected all voxels in the cube into every image, if the image coordinate defines a background pixel, the voxel is marked to be deleted (voting).

Shape from silhouette algorithms intersect all silhouette cones from multiple images to achieve the estimate geometry of the object that is called the visual hull. In Figure 3, for simplicity, the volume constructed is shown in 2D with its input images in 1D. The black polygon shows the visual hull of the object.

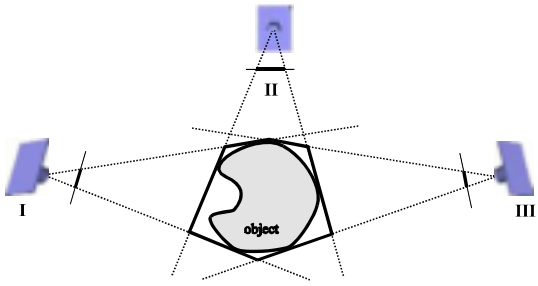

Figure 3: Method of volume intersection.

Shape from silhouette method is a conservative method and it does not carve away the voxels it should not. Thus, visual hull contains the true shape. When the greater number of views is used, this technique progressively refines the object model. However, if only a few views are used the recovered shape can be very coarse since it cannot detect the object concavities. Therefore silhouette contours alone cannot recover the object model geometry precisely and should be supported by another technique to get into the critical, concave areas. See (Kuzu and Rodehorst, 2000) or (Kuzu and Sinram, 2002) for more details.

3.2 Voxel Neighborhood

Before we introduce some tools regarding voxels, we want to clarify the understanding of neighborhood in 3D. When we investigate digital raster images, a pixel has neighbors of two degrees: those connected by an edge and those connected only by a node. They are also called 4- and 8-connected neighborhoods.

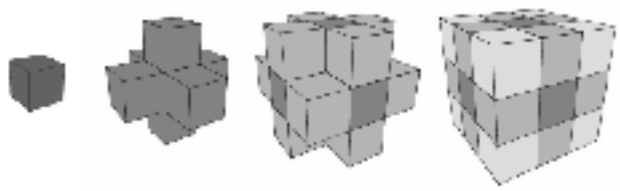

Figure 4: A voxel and its neighbors

In the three dimensional case, a voxel has six neighbors connected by a face, twelve connected by an edge and another eight connected by a node, which results in a total of 26 neighbors. Following 2D images, we call them 6-, 18- and 26-connected neighborhoods. Figure 4 illustrates the different degrees of neighborhood.

3.3 Line Traversal

Line traversal becomes an important tool for various tasks. For example projections of voxels to pixels and vice versa require a line traversal. The basic challenge is to do an operation on each voxel along a line, which is defined by a start- and endpoint.

The first task would be to define a sensible start- and endpoint. The line itself is defined by the image projection center (X_0, Y_0, Z_0) and either a voxel (V_X, V_Y, V_Z) or a pixel (i, k, $-c$). Running any kind of operator on a voxel only makes sense within the defined voxel cube, since there is no information outside. Thus, for the sake of performance, we have to define the line so that it completely encloses the cube, but not much longer.

The simplest limitation might be to look for those two λ-values, so that:

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

will completely enclose the voxel cube for every image point. In order to do so, the eight corners of the cube have to be processed for each image only once. Figure 5a illustrates this method.

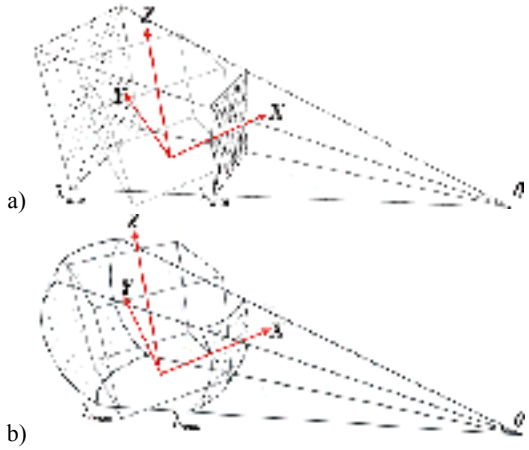

Figure 5: Geometric restriction of the line traversal.

A second approach might be to enclose the cube with a sphere and calculate its two intersection points (if any) with the line of sight. Here, the λ-factor is individual for each pixel, so that we can expect a tremendous loss in performance, outweighing the advantage that the line might be a better approximation. This approach is shown in Figure 5b.

The second line tracing problem is the connectivity of the line. In three dimensions we can either define 6-, 18-, or 24-connected lines. In consequence more or fewer voxels will be traversed.

Although the algorithm for the 18-connected line is easier to implement, the 6-connected line includes some more voxels.

For instance, an 18-connected line could intersect an 18-connected surface without detection. An algorithm for a 6-connected line can be found in (Amanatides and Woo, 1987). A comparison of the two line types is displayed in figure 6.

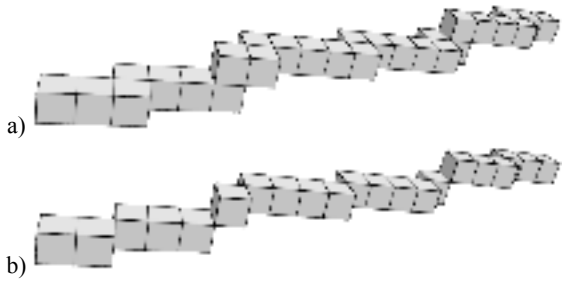

Figure 6: 6- and 18-connected 3D-lines.

The algorithm for the 18-connected line can be described by the following pseudocode:

```
function linetracing (P0=start, P1=end)
{
     let dx = P1.x - P0.x
     let dy = P1.y - P0.y
     let dz = P1.z - P0.z

     let maxlen = MAX[ABS(dx), ABS(dy), ABS(dz)]

     let sx = dx / maxlen
     let sy = dy / maxlen
     let sz = dz / maxlen

     let P = P0

     for (0 … maxlen)
     {
          call operator with P

          P.x = P.x + sx
          P.y = P.y + sy
          P.z = P.z + sz
     }
}
```

3.4 Visibility Information

It is often crucial to find out which voxel is visible in which image. From polygonal models we know that there are two reasons why a point (or a face) is not visible from a certain viewpoint. The first and easy case is when the point is lying on that side which is facing away from the camera. It is only necessary to calculate the triangles normal vector and check if it is pointing towards the camera.
The other case is that a point is occluded by another object in between. The major approach creates a depth buffer, which stores the distances of the recently projected points. Any point further away is just skipped, while the buffer is updated with the nearest point at a time.
As voxel techniques are very different from polygonal models, we cannot simply take over their algorithms. The depth buffer is flexible enough to apply it here as well, but we will use the line tracing to achieve better and faster results.
Now, when we want to learn about the visibility of a voxel in a certain image, a line is defined with the voxel itself as start point, and the image projection centre as end point. We will use the line tracing algorithm to check each voxel along the line, whether it is background (and therefore transparent) or object voxel (opaque). As soon as an opaque voxel is encountered, the initial voxel can be considered occluded. When the line exits the defined voxel cube, it can be stopped, assuming that the voxel is visible. Whether lying on the backside, or occluded by another voxel, the algorithm will correctly tell if the voxel is visible or not.
The basic idea is depicted in Figure 7.

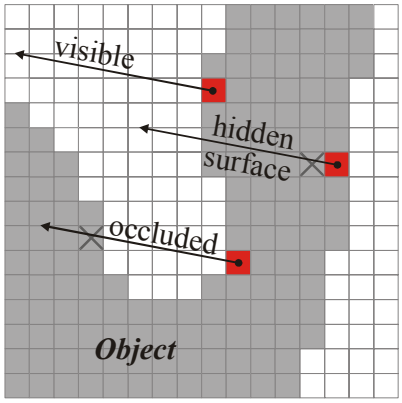

Figure 7: Recovering visibility information

In Figure 11 on the last page the differences of the projection results are displayed, based upon the decision, which voxels are excluded from consideration. It can also be seen, that an indirect transformation (the image pixel looks up the corresponding voxel) delivers by far the best result.

3.5 Surface Normal Vector

The surface normal vector can tell us in which direction a voxel is facing. It is the normal vector of the tangential surface from the voxel of interest. While in polygonal models it is very easy to calculate the normal vector (it is simply the cross product of two sides of a triangle) it becomes more sophisticated with voxel models.
Our basic approach calculates a least-squares adjustment on a specific subset of surface voxels for the plane equation:

$$a \cdot x + b \cdot y + c \cdot z + d = 0 \tag{2}$$

Therefore we define a certain radius within which we take all surface voxels (see figure 8) and write them into the matrix A, with:

$$A = \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \vdots & \vdots & \vdots \end{bmatrix} \tag{3}$$

and the unknown vector:

$$X = \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix} \tag{4}$$

Since only the direction of the tangential surface is of interest, not the translation offset ***d*** (see equation 3), the unknown vector contains only the three elements ***a***, ***b*** and ***c***, which directly correspond to the elements of the surface normal vector.

We now have to solve $\boldsymbol{A}\cdot\boldsymbol{X}=\boldsymbol{0}$, where we are not interested in the most obvious case $\boldsymbol{X}=\boldsymbol{0}$. We will use the singular value decomposition (SVD) to compute the best solution, with $\boldsymbol{X}\neq\boldsymbol{0}$ (Hartley, Zisserman, 2000). The SVD will be used to split the design matrix $\boldsymbol{A}$ into three new matrices $\boldsymbol{U}$, $\boldsymbol{D}$ and $\boldsymbol{V}^T$, such that $\boldsymbol{A}=\boldsymbol{U}\cdot\boldsymbol{D}\cdot\boldsymbol{V}^T$, where $\boldsymbol{U}$ and $\boldsymbol{V}$ are orthogonal matrices and $\boldsymbol{D}$ is a diagonal matrix with non-negative entries. From adequate literature we can learn that the solution to an equation $\boldsymbol{AX}=\boldsymbol{0}$ corresponds to that column of the $\boldsymbol{V}$ matrix, which corresponds to the smallest value in the $\boldsymbol{D}$ matrix.
The recently derived unknown vector $\boldsymbol{X}$ directly contains the elements of the desired normal vector, so that $\boldsymbol{X}=\boldsymbol{n}$.

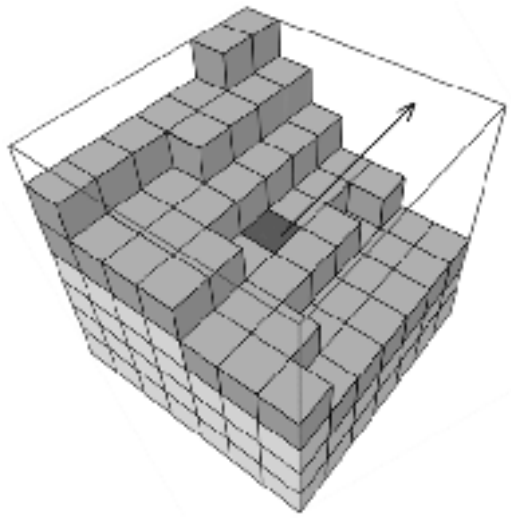

Figure 8: Surface normal vector from a 7^3 section of a cube.

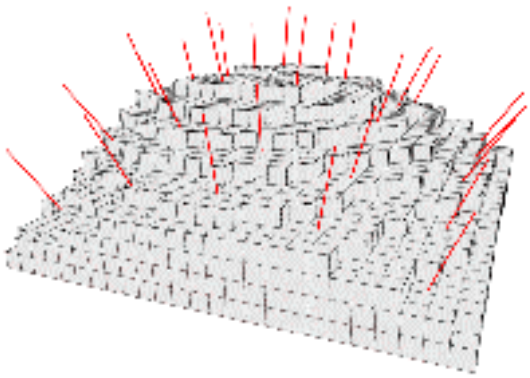

Figure 9: Surface normal vectors on a larger selection.

The surface normal vector gives information of the direction, where a voxel is facing. So it can serve as a quality measure for visibility. When we simply calculate the angle between the surface normal and the ray of sight, it can tell us whether the voxel is 'looking in our direction'. Hence, if the angle is small, it is facing the image, and if it exceeds 90° it can be considered hidden.

4. REFINEMENT ALGORITHM

As stated above, the shape from silhouette does not always recover the true surface of the object. Therefore, in those regions with a false surface we need to refine the carving.
As a fact, only true surface points will be projected into corresponding image points, assuming correctly oriented images are provided.
Thus, if we project any surface voxel into two or more images, which are most likely to have the best view of this voxel, they should be considered similar with an appropriate similarity operator. The best images can be chosen upon the visibility information and the surface normal vector, described in the last chapter.
Consequently, if the set of pixels or pixel regions respectively, are proven to be different, we can assume that the corresponding voxel is not part of the true surface.
However, if the voxel projects into homogeneous regions of the image, the similarity operator will constantly return a high similarity value. There is no way to tell the correct correspondence here.
Consequently, we can classify each surface voxel into three classes: surface voxel, non surface voxel, uncertain. Once a voxel has been considered surface voxel it should remain fixed and should not be evaluated again. Non surface voxels on the other hand will be erased and the voxels underneath become surface voxels and will be considered in another iteration.
We can make use of line tracing when we encounter a non-surface voxel. We define a reference image, which will result in a line from the images projection centre to the voxel. Now we can trace this line, starting with the voxel, away from the image into the object. For each voxel along this line, we calculate a new similarity value for the updated image projections. Where we find the maximum similarity exceeding a sensible threshold, we can assume it as the true surface and classify the encountered voxels accordingly.

5. SUMMARY

In this paper we presented several powerful tools which are useful in voxel based reconstructions. An experimental image acquisition setup was explained on which the introduced algorithms were tested. The shape from silhouette method was briefly explained since it was introduced in earlier works.
Upon this approximated reconstruction, the line traversal is applied in many occasions such as visibility computation, texture mapping, similarity calculation. As an extension to visibility information, we introduced the surface normal vector derived from a regional section of surface voxels. All these methods are sensitive to the degree of neighborhood, which has been introduced in detail.
We explained how the refinement algorithm makes use of these tools in order to improve the initial approximate reconstruction.

REFERENCES

Amanatides, J., Woo A., 1987. A Fast Voxel Traversal Algorithm for Ray Tracing. *Proc. Eurographics '87,* pp 1-10.

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

Hartley R., Zisserman A., 2000. *Multiple View Geometry in Computer Vision.* Cambridge University Press.

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

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

Kuzu, Y. Sinram, O., 2002. Photorealistic Object Reconstruction using Voxel Coloring and Adjusted Image Orientations. *ACSM/ASPRS Annual Conference*, Washington DC, Proceedings CD-ROM, Proceed\00437.pdf.

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.

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

a) Original photograph b) Visualization of a 256^2 cube c) Visualization of a 512^3 cube d) Lighting simulation based upon surface normal vector

Figure 10: Visualization of a replica of a Greek vase, Geometrical Period 900-800 B.C.

a) Projection of all voxels into an image b) Projection of all surface voxels into an image c) Projection of only visible voxels into an image. d) Image generated by ray tracing (image to voxel)

Figure 11: Comparison of different projection methods on the model of Nefertiti.