Complex 3D Shape Recovery Using a Dual-Space Approach

Chen Liang and Kwan-Yee K. Wong
Department of Computer Science, University of Hong Kong,
Hong Kong SAR, China
{cliang, kykwong}@cs.hku.hk

Abstract

This paper presents a novel method for reconstructing complex 3D objects with unknown topology using silhouettes extracted from image sequences. This method exploits the duality principle governing surface points and their corresponding tangent planes, and enables a direct estimation of points on the contour generators. A major problem in other related works concerns with the search for a tangent basis at singularities in the dual tangent space. This problem is addressed here by utilizing the epipolar parameterization for identifying a well-defined basis at each point, and thus avoids any form of search. For the degenerate cases where epipolar parameterization breaks up, a fast on-the-fly validation is performed for each computed surface point, which consequently leads to a significant improvement in robustness. As the resulting contour generator points are not suitable for direct triangulation, a topologically correct surface extracting method based on slicing plane is presented. Both experiments on synthetic and real world data show that the proposed method has comparable robustness as those existing volumetric methods regarding surface of complex topology, whilst producing more accurate estimation of surface points.

1. Introduction

In an environment of unknown lighting condition, the 3D shape of a textureless object could hardly be recovered using point features or shading information. Silhouettes, in this case, can often be reliably extracted and provide much richer information for shape estimation. Unlike point features, silhouettes are viewpoint dependent, and traditional stereo vision techniques do not suffice for computing the depth of the surface points. This urges for the development of alternative approaches for computing the 3D structure indirectly.

Each silhouette, if treated as one or several non-intersecting curves, defines a family of tangent planes of which the envelope is the *viewing cone* [4]. The envelope of tangent planes defined by a set of cameras resembles the visual hull bounding the object surface with respect to the set of cameras. In the literature, it has been shown that given the knowledge of either points or tangents, their dual counterpart could be estimated. This infers that it is possible to estimate a surface in the original space from the dual tangent space. In practice, in order to estimate the surface from its discretely sampled tangent planes, it is necessary to identify a well-defined tangent basis in the dual tangent space. This is, however, not a trivial task due to the singularities arising from bi-tangents in the original space.

One important contribution of this paper is that we utilize the *epipolar parameterization* [7] for selecting a subset of neighbors in the dual space for the estimation of tangent basis. This does not require any search in the dual space as previous methods do. Hence our estimation of the tangent basis in the dual space is unaffected by aforementioned singularities and can be done in constant CPU time for each point. As shown in the experiments, our method is capable of recovering complex shapes with non-zero genus, which often imply singularities in the dual space. Our method also takes into account the extreme cases exhibited in epipolar parameterization and detects low quality estimations on-the-fly. These significantly increase the robustness of our algorithm and naturally extend our method to handle shapes with sharp edges.

Another contribution of this paper is the introduction of a topological preserving surface extraction algorithm. As limited by the number of cameras, the recovered points are usually unevenly distributed along the two epipolar parametric directions. A direct triangulation, in this case, will produce ill-formed triangles. We present a slice based approach for surface generation which produces a robust reconstruction.

1.1. Related Works

The dominant approach for reconstruction from silhouettes is the volumetric approach. The concept of using vol-

ume intersection was first proposed by Martin and Aggarwal [10], who introduced *volume segment* under orthogonal projection. In [8], the volume intersection was conceptualized as the *visual hull*, which gives a bounding volume for model reconstruction from silhouettes. The OcTree [6] and its variants [11, 13] adopt a regular grid or *voxels*, and provide a more efficient data structure for representing the visual hull volume. These approaches usually make no assumption on the surface topology and are robust to complex objects. The quality of the approximation to the visual hull, however, is limited by the chosen size of the voxels.

In contrast to the volumetric approach, surface-based approaches often attempt to explicitly compute the points lying on the surface or visual hull. In [1], Baumgart pointed out that the visual hull could be computed as the polyhedral intersection of the viewing cones. Later on, Boyer and Berger [3] showed that depth along the visual ray could be computed through fitting an osculating quadric to its two epipolar correspondences. Recently, Lazibnik et al. [9] derived the visual hull as a generalized polyhedron computed from the epipolar constraints, avoiding both direct polyhedral intersections and voxel carving. Both approaches, however, do not guarantee to work given an object with non-zero genus. In [4], Boyer and Franco proposed a hybrid method that combines the advantages of both volumetric and surface-based approaches. Their approach extended the idea in [10] to segment the visual rays, and the resulting points are tetrahedrized into irregular cells. The final surface is extracted from those cells having the projections within all silhouettes. They showed that their method can handle more complicated objects than the surface-based approaches, while producing results that are more accurate than the volumetric approaches. However, the quality of the final model does not always increase with the number of cameras/images used.

Brand et al. [5] recently proposed a method complementing the literature of volumetric and surface-based approaches. They exploited the *duality principle* exhibited between the surface and the tangent envelope defined by the viewing cones, and computed a solution with minimum algebraic error for each point from a non-singular set of neighbors in the dual space. Their approach, however, relies on proximity in the dual space. Searching for a non-degenerate set of neighbors is not trivial, especially when the surface geometry is complex. They also appear not having explicitly stated their method for extracting the surface from the computed point cloud which, in most cases, has uneven distribution due to the discrete nature of the image sequence.

In this paper, we extend the use of duality theory and present a complete and practical framework for shape recovery from silhouettes. This framework aims at robustly recovering complex shape with non-zero genus.

2. Theoretical Framework

We shall consistently represent scalar values in italic font k, matrices in uppercase bold font $\mathbf{P}$, and column vectors (points, tangent vectors, etc.) in lowercase bold font $\mathbf{r}$. The homogeneous counterpart of a vector is written as $\widetilde{\mathbf{r}}$, and will be used interchangeably with its non-homogeneous counterpart depending on the context. The dual counterpart of a vector is represented by appending a superscript asterisk $\mathbf{r}^*$.

2.1. Geometry of the Surface

A surface is locally parameterized as vector valued function $\mathbf{r}(s,t)$ and is assumed $\mathcal{C}^2$ continuous. The tangent plane at each surface point is a 4-vector denoted as $\mathbf{t}(s,t)$.

Let us consider a calibrated camera. The silhouette in each image is extracted as one or more non-intersecting planar curves delimiting the inside and outside of the silhouette. Each silhouette point $\mathbf{w}$ defines a visual ray from the camera center which is tangent to the surface. The visual rays of this silhouette collectively form a viewing cone. The locus of their tangent points is a spatial curve often termed as the *contour generator* [7]. Suppose the silhouette curve is at least $\mathcal{C}^1$ continuous, each silhouette point $\mathbf{w}$ has a well-defined tangent line. This tangent line back-projects to a tangent plane touching the object surface and contains the visual ray defined by $\mathbf{w}$. Intuitively, the tangent plane envelope of the whole silhouette curve is the viewing cone defined by this silhouette (see Figure 1).

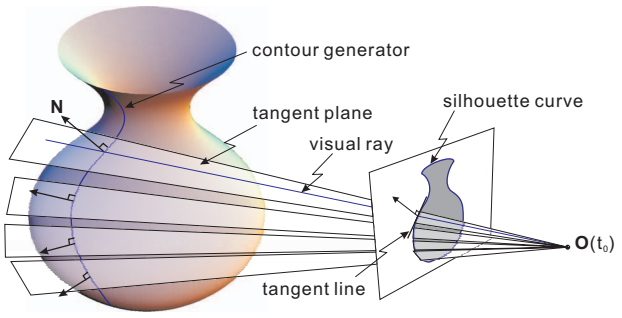

Figure 1. Geometry of the surface

Given a finite number of cameras with distinct viewpoints, the tangent planes defined by all the silhouettes form a non-degenerate tangent plane spaces in $\mathbb{R}^3$. The envelope of these planes is equivalent to the intersection of all the viewing cones, i.e., the visual hull. In the next section, we present an efficient method motivated by the principle of duality for recovering points along the contour generators.

2.2. Theory of Duality

In 3D space, a unified treatment for points and planes is to represent them in homogeneous form as 4-vectors. Under this homogeneous coordinate system, the principle of duality suggests that for all theorems governing points and planes, the roles played by points and planes are interchangeable. A surface point $\mathbf{r}$ and the tangent plane $\mathbf{t}$ at that point form a dual pair, satisfying the following symmetric relationship:

$$\widetilde{\mathbf{r}}^\top \mathbf{t} = \mathbf{t}^\top \widetilde{\mathbf{r}} = 0 \tag{1}$$

As such, the tangent plane vector $\mathbf{t}$ is conveniently denoted as the dual of $\widetilde{\mathbf{r}}$, i.e., $\widetilde{\mathbf{r}}^* \triangleq \mathbf{t}$. Note that $\widetilde{\mathbf{r}}^*$ can also be treated as a 'point' in the *dual space*. In this sense, the dual to $\widetilde{\mathbf{r}}(s,t)$ is a manifold in the dual space $\widetilde{\mathbf{r}}^*(s,t)$.

We know that given the parametric representation of a surface, it is trivial to compute the tangent plane to the surface at a given point. Motivated by the principle of duality, if the roles played by points and planes are interchanged, the original surface could also be computed from its tangent planes as well. A mathematical proof of such a dual relationship in 2D has been derived in [5]. Now we consider the 3D case directly. In 3D case, the normal of the tangent plane at a point is orthogonal to the tangent vector along the two parametric directions. $\widetilde{\mathbf{r}}^*(s,t)$ is therefore computed as:

$$\widetilde{\mathbf{r}}^*(s,t) \propto [\widetilde{\mathbf{r}}_s(s,t), \widetilde{\mathbf{r}}_t(s,t), \widetilde{\mathbf{r}}(s,t)]^\perp \tag{2}$$

where $\mathbf{r}_s(s,t)$ and $\mathbf{r}_t(s,t)$ are the partial derivatives of $\mathbf{r}$ with respect to s and t respectively. We now interchange the roles of $\mathbf{r}(s,t)$ and $\mathbf{r}^*(s,t)$. The left hand side of (2) should become the point on the original surface. This observation is formally described in the following proposition:

Proposition 1. *Assume that the tangent plane at each point of a $\mathcal{C}^2$ continuous surface is known. Points on the surface could be computed as:*

$$\widetilde{\boldsymbol{r}}(s,t) \propto [\widetilde{\boldsymbol{r}}^*_s(s,t), \widetilde{\boldsymbol{r}}^*_t(s,t), \widetilde{\boldsymbol{r}}^*(s,t)]^\perp \tag{3}$$

In practice, the tangent plane space is obtained from a discrete set of cameras. It is thus impossible to obtain an implicit representation for the dual manifold $\widetilde{\mathbf{r}}^*(s,t)$. The tangent basis for a point on the dual manifold, namely $\mathbf{r}^*_s(s,t)$ and $\mathbf{r}^*_t(s,t)$, is often approximated from neighboring points using finite differences. This implies that the quality of the estimation strongly depends on how we select those neighboring points. Unfortunately, this task is not at all trivial and poses a major difficulty in some previous approaches, for the following reasons:

- Two points $\widetilde{\mathbf{r}}_1$ and $\widetilde{\mathbf{r}}_2$ in the original space sharing a common tangent plane (bi-tangency) will be mapped to the same point $\widetilde{\mathbf{r}}^*$ in the dual space, which implies a singularity;
- The tangent space $\widetilde{\mathbf{r}}^*$ computed by evenly sampling the silhouettes may still be evil-distributed. It is very difficult to set a distance threshold by which a nearby point could be qualified as a proper neighbor.

Hence for discretely sampled dual manifold, it is not reliable if we solely use distance measure for selecting neighbors. In [5], the authors tried to limit the search among the dual space points computed from immediate neighboring views. Doing this has indeed decreased the chance of picking bad neighbors. However, it still relied on the distance measure, which means it may still be puzzled by the singularities and evil-distribution of the dual space points.

In our approach, we rely on the epipolar structure to locate the neighbors in the dual space, which in fact does not require any knowledge about the actual surface. As we do not search in the dual space, we are unaffected by the aforementioned evil-distribution and singularities in the dual space.

2.3. Epipolar Parameterization

Assume the object surface is $\mathcal{C}^2$ continuous. Epipolar parameterization gives a well-defined local neighborhood for each point on the surface. An elaborated definition of epipolar parameterization can be found in [7], and here we recast it into our framework. A point $\mathbf{r}$ on the surface is given as:

$$\mathbf{r}(s,t) = \mathbf{O}(t) + \lambda(s,t)\mathbf{p}(s,t) \tag{4}$$

where s is the curve parameter along contour generators, t is the time parameter, $\mathbf{O}(t)$ is the center of the camera at time t, $\mathbf{p}(s,t)$ is the viewing vector from $\mathbf{O}(t)$ to the focal plane at unit distance for the point $\mathbf{r}(s,t)$, and $\lambda(s,t)$ is the depth of the point $\mathbf{r}(s,t)$ along the optical axis from $\mathbf{O}(t)$.

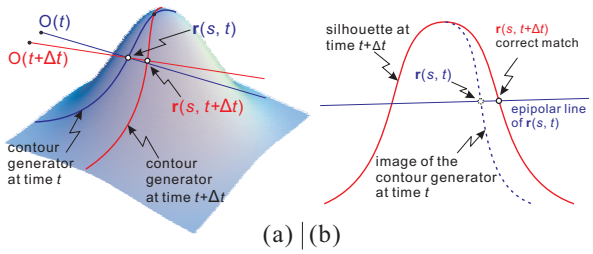

Figure 2. Epipolar parameterization and matching

Fixing the s and t parameters respectively, we result in two families of curves forming a coordinate grid on the surface, i.e., $\mathbf{r}(s,t_0)$ and $\mathbf{r}(s_0,t)$. The curve represented by

$\mathbf{r}(s, t_0)$ can be viewed as the contour generator that corresponds to the view t_0. The curve $\mathbf{r}(s_0, t)$ can be taken as the locus of points on the surface grazed by a visual ray from the camera center which is moving over time. In a discrete sense, this curve is formed by joining the correspondent points, i.e., with same s, on successive contour generators. To uniquely define such a correspondence, the matching point $\mathbf{r}(s_0, t+\Delta t)$ of $\mathbf{r}(s_0, t)$ is found by intersecting the contour generator at $t + \Delta t$ with the epipolar plane defined by $\mathbf{r}(s_0, t)$, $\mathbf{O}(t)$ and $\mathbf{O}(t + \Delta t)$ (see Figure 2(a)). Reflected on the image plane at time $t + \Delta t$, $\mathbf{r}(s_0, t + \Delta t)$ is found along the epipolar line of $\mathbf{r}(s_0, t)$ where it intersects the silhouette curve (see Figure 2(b)).

Under the epipolar parameterization, each point on the contour generator has a well-defined neighborhood. As we have been able to map any point on a contour generator to its dual (see section 2.1), the neighborhood information on the dual manifold can be directly inferred from the neighborhood information granted by the epipolar parameterization.

Compared with the work of Brand et al. [5], our method relies on the epipolar structure in the original space for the desired neighborhood information. Hence it is not affected by bi-tangency, which only causes singularity in the dual space. This allows our method to handle much more complicated surface with non-zero genus. In addition, doing epipolar matching requires simply 2D intersection computation and does not need to search in the dual space, making the method more efficient in terms of computational time.

It is noted that the epipolar parameterization may break up in a few extreme cases [12]. These cases, together with the countermeasures adopted in our implementations, are discussed in section 3.1.

3. Estimation of points on the surface

Active B-Snakes [7] are adopted to extract silhouettes from images. Each silhouette consists of one or more closed B-Spline curves. The advantages of using B-Spline include (1) sub-pixel precision, (2) trivial computation of the tangent at each point on the silhouette, and (3) close-formed solution for epipolar line intersection. Furthermore, we can make assumption on the sides of a silhouette with respect to the parametric direction of the B-splines. This lead to a simple and uniform treatment of silhouettes consisting of an arbitrary number of B-Splines.

Each surface point is independently estimated from its dual space neighbors. Conveniently, we could achieve minimum memory consumption by collecting only the necessary dual space neighbors per estimation. Let $\mathbf{r}(s, t)$ be the point we want to estimate. This point lies on the contour generator at time t. We denote the projection of this point as $\mathbf{w}(s)_t$, which should lie on the silhouette curve at time t. By back-projecting the tangent line at $\mathbf{w}(s)_t$, we obtain the tangent plane, i.e., the dual space point $\mathbf{r}^*(s, t)$. Now we need to collect the dual space neighborhood along two parametric directions of the epipolar parameterization. The computation of $\mathbf{r}^*(s \pm \delta s, t)$ is straight-forward: they are the back-projections of the tangent lines at $\mathbf{w}(s \pm \delta s)_t$. The computation of $\mathbf{r}^*(s, t \pm \delta t)$ is done by epipolar matching on the neighboring views ($t \pm \delta t$), operating purely on 2D image planes: (1) map $\mathbf{w}(s)_t$ to the epipolar lines in view $t - \delta t$ and view $t + \delta t$, (2) compute the intersections between the epipolar lines and the silhouette in each of these views, (3) determine the best epipolar match among the intersections in each view. The tangent planes associated with these epipolar matches are then taken as $\mathbf{r}^*(s, t \pm \delta t)$. Since silhouettes are represented by B-Splines, the intersections can be accurately computed as close-formed solutions.

Once the dual space neighbors for $\mathbf{r}^*(s, t)$ have been collected, we can apply Proposition 1 to compute $\mathbf{r}(s, t)$, which is essentially the estimation of a tangent plane from a set of neighboring points. In our implementation, we simply rearranged equation (3) in Proposition 1 into a system of linear equations. This treatment allows us to conveniently incorporate a visual ray constraint described in 3.1.

Note that while the above process suffices in most cases, some special care needs to be taken for a few extreme cases of epipolar parameterization.

3.1. Handling extreme cases

We currently consider two common extreme cases. The first case concerns with cylindrical surface (see Figure 3 (a)). The contour generator resembles a straight-line, thus the tangent planes along the contour generator do not change as parameter s changes. Accordingly, the null-vector of the linear system obtained from equation (3), ideally, also spans along a straight-line. A stronger constraint is needed in order to fix the estimated position along this line. A natural choice is to incorporate the visual ray constraint, which is given by:

$$\widetilde{\mathbf{w}}(s)_t = \mathbf{P}_t \widetilde{\mathbf{r}}^{**}(s, t) \tag{5}$$

where $\widetilde{\mathbf{w}}$ is the image position in homogenous form, $\mathbf{P}_t$ is the projection matrix of view t, and $\widetilde{\mathbf{r}}^{**}(s, t)$ is the contour generator point having the image position $\widetilde{\mathbf{w}}$. This constraint could also be rearranged into two linear equations. They are directly incorporated into the system of linear equations mentioned earlier, and impose only a negligible extra computational cost.

Another extreme case is caused by self-occlusion on the surface (see Figure 3(c)). A typical indication is the $T - junction$ (see Figure 3(d)). In such a case, the correct epipolar match for a point does not lie on the silhouette and thus the correct match could not be identified. However, in practice, we may not be able to differentiate this case and

may pick a wrong epipolar match. Given that we have enforced the visual ray constraint, these wrong neighbors, in the worst case, cause the point to fly-out of the visual hull in the direction of the visual ray.

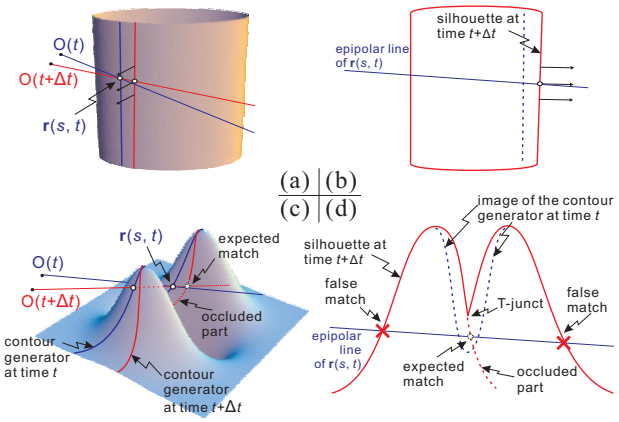

Figure 3. Extreme cases in epipolar parameterization

We introduce an on-the-fly test for detecting this kind of fly-out points. It is an image consistency test complying to the visual hull concept: the point must lie on or inside all the silhouettes. This test could be efficiently carried out: we rasterize the silhouettes into binary maps of user-specified resolution. Each estimated point is then projected onto the integral coordinates of these binary maps. If the binary values at all projections do not unanimously indicates 'inside', the point is not a good estimation.

The points disqualified by the above test are singled out and may subject to other estimation methods. In the current implementation, we simply drop them as they only account for a very small portion of all the estimations and have little impact on the model we should have obtained.

4. Extraction of the target surface

The result of the earlier estimation are points lying on the contour generators, each of which coupled with the tangent plane at that point. This means that the surface normal at each point is known. Unfortunately, these points are not distributed uniformly. This is due to the fact that the sampling density along each contour generator can be set arbitrarily high, while the density across the contour generators is limited by the number of distinct viewpoints. Direct triangulation, in this case, results in ill-formed triangles. On the other hand, unlike unorganized point cloud, we do know the connectivity of points along the same contour generator and also the spatial order of the contour generators. We therefore propose a simple but robust method for extracting the surface with the connectivity and ordering information available.

4.1. Slicing the contour generators

The proposed surface extracting method is built on a slice based re-sampling method, and it targets at producing more evenly distributed mesh grids. The earlier computed contour generators are now re-sampled by parallel slicing planes. Each slicing plane contains the intersections between the contour generators and the sampled plane. The normal of each point is interpolated from the normals of the adjacent points on the same contour generator. For a better visual effect, these slices are parallel to the plane maximally spanning all the camera centers.

Once the re-sampled points on each slice are obtained, we proceed to connect these points and form 2D polygons. We first link these points according to the spatial order of the corresponding contour generators. This results in one polygon on each slice. For complex shapes, this single slice polygon may not reflect the true topology of the surface. We need to break this polygon and regroup the points into several smaller polygons complying to the topology suggested by the silhouettes. To do this, we project each edge of this polygon onto the silhouettes and break the edge if it falls partially outside of any silhouette. The resulting partially connected polygonal vertices reform new polygons which should now project within all the silhouettes. The same process is repeated for each slice.

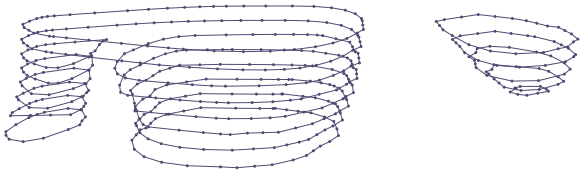

Figure 4. 2D polygons on the slicing planes generated from a David turn-table sequence

Note that the if we link the contour generator points on each slice directly, we result in a shape generally smaller than the visual hull volume. To create a better visual effect, we fit a smooth second order parametric curve $\mathbf{x}(t)$ on the slicing plane replacing long edges of the polygons on each slice. An example of subdivided and smoothed slices is given in Figure 4.

4.2. Generating final surface

The final object surface is extracted from the polygons on each slicing planes using some well-established method.

We borrow and modify a method described in [2]. The surface normal at each vertex is simply the interpolated normal from the previous slicing step (see section 4.1).

The overall algorithm outline for extracting the surface from the contour generator points is given below:

Algorithm 1 Extracting object surface

for each contour generator Γ_i **do**
  for each connected segment along Γ_i **do**
    if has intersection p with slicing plane π_k **then**
      add p to pi_k along with the interpolated normal
    end if
  end for
end for
for each slicing plane π_k **do**
  correct topology and do sorting for the points on π_k
  fit smooth curve if user specified
  generate surface triangles with points on π_{k-1}
end for
map texture for each triangle

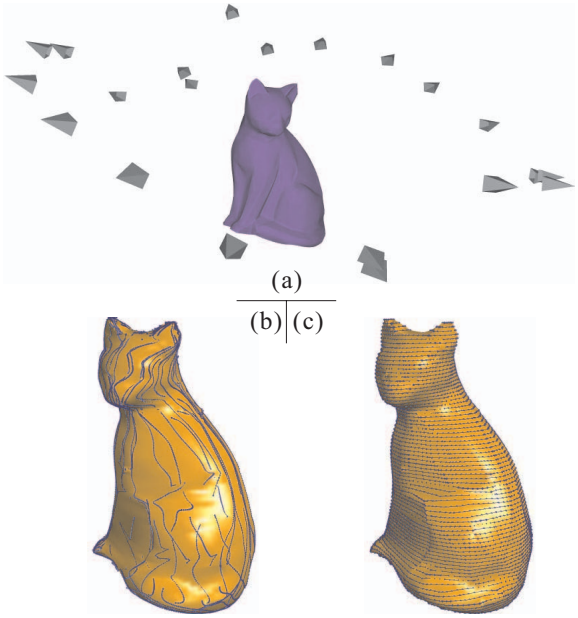

Figure 5. Reconstruction of a Cat model (a) The original model and the camera positions; (b) Estimated contour generators super-imposed on the surface (c) Slice polygons super-imposed on the surface

5. Experiments

Our approach has been extensively tested on both synthetic and real world data. Three representative experiments are shown in this section. The first experiment is based on 20 images of a synthetic cat model. The projection matrices used for rendering are already known (see Figure 5(a)). The camera centers are purposely set to irregular positions around the object. An implied requirement by epipolar parameterization is that each camera is not too far from its neighboring cameras.

The second experiment (see Figure 7) is performed over a turntable sequence of 20 images. All the cameras have been calibrated. We compare the result of a volumetric approach [13] with that of our algorithm. The surface extracted from the OcTree using marching cube algorithm has severe bumpy effect (see Figure 7(c)), even it consists of more than 1.7k triangles. By smoothing the extracted surface, the bumpy effect could be alleviated, but the details of the surface are also smoothed out (see Figure 7(d)). The model (see Figure 7(e)) generated by our approach contains only 7k triangles. The details and the sharp edges are much more well-preserved than using the volumetric approach.

The third experiment (see Figure 6) is conducted on another turntable sequence, with fairly complex surface geometry. The experiment shows that our method can robustly recover the surface with non-zero genus, even when we do not have any assumption on the topology of the surface.

Figure 6. Reconstruction of a David statuette from a turntable sequence (20 images)

6. Discussion

In this paper, we have presented a novel method for recovering 3D shape with unknown topology from silhouettes extracted from 2D image sequence. This method exploits

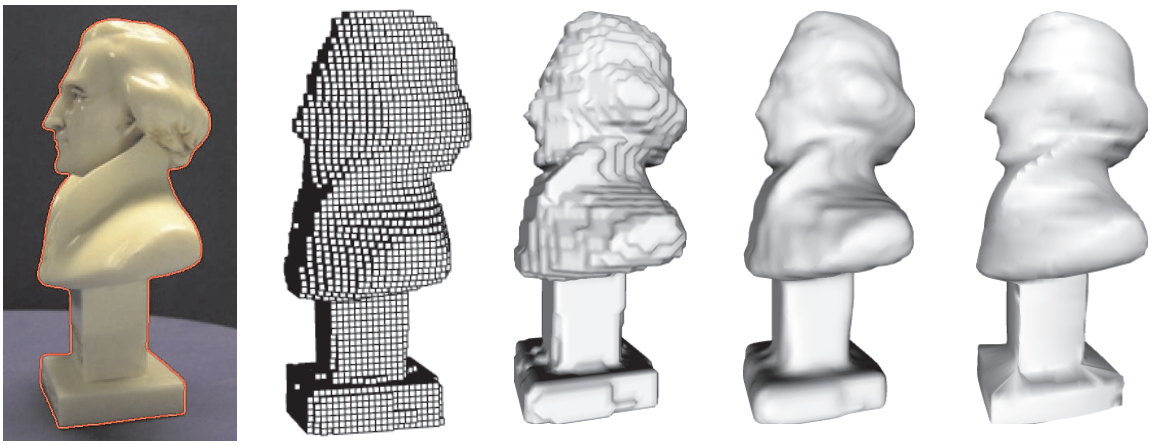

Figure 7. Reconstruction of a Jefferson statuette from a turntable sequence (20 images), compared with the volumetric method; (a) The target object; (b) Reconstructed OcTree; (c) Surface extracted from the OcTree (1.7k triangles) (d) Smoothed surface by averaging neighboring vertices in (c); (e) Surface obtained with our approach (7k triangles, 60 slices);

the dual relationship between the target surface and the tangent plane space sampled from the silhouettes, and recovers surface points on the contour generators directly. The use of epipolar parameterization allows the method to handle complicated shape robustly. We have also proposed a slice based approach to produce a topologically correct surface representation from the surface points recovered.

One limitation of the current approach is that we assume that a contour generator is a single continuous spatial curve. However, extreme cases such as self-occlusion may lead to discontinuities. We are seeking for possible ways that handle these cases explicitly. There are also rooms for improvement in the surface extraction method. Despite its simplicity and robustness, current method re-samples the surface with even-spaced slices, and sharp features parallel to the slicing planes may be missed.

Acknowledgments

This project is supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China, under Project HKU 7155/03E.

References

[1] B. Baumgart. A polyhedron representation for computer vision. In *AFIPS National Computer Conference*, 1975.

[2] J.-D. Boissonnat. Shape reconstruction from planar cross sections. *Computer Vision, Graphics and Image Processing*, 44(1):1–29, October 1988.

[3] E. Boyer and M.-O. Berger. 3d surface reconstruction using occluding contours. *International Journal of Computer Vision*, 22:219–233, 1997.

[4] E. Boyer and J.-S. Franco. A hybrid approach for computing visual hulls of complex objects. In *Computer Vision and Pattern Recognition*, volume 1, pages 695–701, Madison, Wisconsin, June 2003.

[5] M. Brand, K. Kang, and D. Cooper. An algebraic solution to visual hull. In *Computer Vision and Pattern Recognition*, volume 1, pages 30–35, Washington, DC, July 2004.

[6] C. Chien and J. Aggarwal. Volume/surface octrees for the represntation of three-dimensional objects. *Computer Vision, Graphics and Image Processing*, 36(1):100–113, October 1986.

[7] R. Cipolla and A. Blake. The dynamic analysis of apparent contours. In *International Conference on Computer Vision*, pages 616–623, Osaka, Japan, December 1990. IEEE Computer Society Press.

[8] A. Laurentini. The visual hull concept for silhouette-based image understanding. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 16(2):150–162, Feburary 1994.

[9] S. Lazebnik, E. Boyer, and J. Ponce. On computing exact visual hulls of solids bounded by smooth surfaces. In *Computer Vision and Pattern Recognition*, volume 1, pages 156–161, Hawaii, December 2001.

[10] W. Martin and J. Aggarwal. Volumetric descriptions of objects from multiple views. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 5(2):150–158, 1983.

[11] R. Szeliski. Rapid octree construction from image sequences. In *CVGIP: Image Understanding*, volume 58, pages 23–32, July 1993.

[12] R. Weiss. The epipolar parametrization. In *Object Representation in Computer Vision, Int. NSF-ARPA Workshop*, pages 101–107, New York City, NY, USA, December 1994.

[13] K.-Y. Wong and R. Cipolla. Structure and motion from silhouettes. In *International Conference on Computer Vision*, volume 2, pages 217–222, Vancouver, Canada, 2001.