3D Reconstruction Using Silhouettes from Unordered Viewpoints

Chen Liang * Kwan-Yee K. Wong

Department of Computer Science, The University of Hong Kong, Pokfulam, Hong Kong SAR, China

Abstract

In this paper, we present a novel approach for reconstructing an object surface from its silhouettes. The proposed approach directly estimates the differential structure of the surface, and results in a higher accuracy than existing volumetric approaches for object reconstruction. Compared with other existing differential approaches, our approach produces relatively complete 3D models similar to volumetric approaches, with the topology conforming to what is observed from the silhouettes. In addition, the method neither assumes nor depends on the spatial order of viewpoints. Experimental results on both synthetic and real world data are presented, and comparison is made with other existing approaches to demonstrate the superiority of the proposed approach.

Key words: 3D reconstruction; dual space; delaunay triangulation.

1 Introduction

Silhouettes are the outlines of real world objects appearing on the images. They provide rich information for digitizing these objects into high quality 3D models. Reconstruction algorithms based on silhouette information are often capable of producing relatively complete 3D models. They can be classified into two main schools according to the metaphor they assume about the object. One school, namely the volumetric school, is characterized by treating objects as solid volumes. This concept first appeared in Martin and Aggarwal [15] where the space is rasterized into parallelogram structure. In a subsequent

* Corresponding author. Tel.: +852 2857 8454
Email addresses: cliang@cs.hku.hk (Chen Liang), kykwong@cs.hku.hk (Kwan-Yee K. Wong).

work, Chien and Aggarwal [5] adopted octree as a more generic representation for the object volume. Potmesil [17] extended the volumetric method to perspective projection with arbitrary viewpoints. The object volume is reconstructed by computing the intersection between the visual cones associated with all viewpoints, resulting in what is often termed the *visual hull*[12]. However, the proposed method needs to create one octree for each viewpoint. Szeliski [18] proposed an efficient method that hierarchically builds up and maintains a single octree. In [9], García and Brunet addressed the problem of weakly calibrated cameras and constructed an octree up to a projective transformation of the scene. In recent works including [7,11], extra information, such as shading information, is also adopted for further refining the initial volumetric visual hull. A major advantage of using a volumetric description is its robustness regarding object with complex topology. However, the accuracy of the reconstruction is limited by the chosen size of the voxel cells. Smaller cells lead to higher accuracy, but at the same time, they also cause the model complexity to increase cubically.

The other school, namely the differential school, treats the object surface as an infinitesimally thin shell. It tends to directly recover the differential properties of the object surface by estimating the contour generators (a.k.a. extremal boundaries or rims). Related work was pioneered by Giblin and Weiss [10] under orthogonal projections and was extended to perspective projections by Cipolla and Blake [6]. A parallel attempt by Vaillant in [19] showed that curvature and depth along a contour generator can be computed given a triplet of continuous viewpoints. Later in [2], Boyer and Berger gave a closed-form solution for estimating the global shape of the object. Their method no longer requires reconstruction planes as in [6] and [19]. However, these methods generally do not take into account objects with non-zero genus. Recently in [4], Brand et al. introduced a very simple algebraic solution based on the *principle of duality* for directly recovering points on the object surface from the dual space points sampled from the silhouettes. A key step of their method is to obtain reliable tangent bases in the dual space. However, their method relies on exhausting the dual space directly, which is often perplexed by singularities caused by bi-tangents on the object surface. Consequently, it is rather difficult for their method to deal with more complicated surfaces. In our previous work [14], we tackled this problem by integrating the *epipolar parameterization* [6] for identifying tangent bases reliably, and avoided any direct search in the dual space. This allows the reconstruction of objects with more complicated shapes and non-zero genus.

There are also hybrid methods that exploit the advantages of both schools. In [13], Lazebnik et al. proposed a *rim mesh* to represent the topology of contour generators. The order of the viewpoints is considered explicitly in the form of rim ordering criterion, allowing the input viewpoints in an arbitrary order. In [3,8], the authors proposed a method (EPVH) to locally join carved visual ray

segments to form a water tight surface. The method has significantly improved over octree based approach in accurately reconstructing the visual hull. However, the reconstructed points are only guaranteed to lie on the visual hull at best, and the visual hull is different from the real object surface. In addition, the difficulty of joining visual ray segments scales fast with the number of views and the complexity of the object.

Generally speaking, differential methods champion volumetric ones in terms of accuracy. However, they lack the robustness of volumetric methods in dealing with variance in object shape and topology. As a result, the reconstructed models are often incomplete and corrupted. In addition, for differential methods, there is often an implicit assumption on the smoothness of the viewer motion, as this is essential for approximating the first and second order differential properties of the object surface. Very few existing works have explicitly considered a more general configuration consisting of unordered viewpoints.

The main contributions of this paper are in two aspects. First we recast the concept of duality appeared in [14] into a general setting consisting of discrete viewpoints with arbitrary spatial order, and introduce a solution for estimating surface points using the information offered by all nearby viewpoints. Another main contribution of this paper is a generic and robust algorithm for generating relatively complete surface meshes from the earlier estimated surface points. This algorithm is extremely useful for objects with unknown and complicated topology. Throughout this paper, we will show the results of both synthetic and real world data, as well as comparison with other existing methods.

This paper is organized as follows. In Section 2, we review the background theories of the dual space method for reconstruction from silhouettes. In Section 3, we introduce our solution for extending the dual space method to handle image sequences of unordered viewpoints. Section 4 proposes a surface extraction algorithm for producing a surface mesh maximally complying with the topology suggested by the silhouettes. Experiment results on both synthetic and real world image sequences are shown in Section 5, and conclusion is given in Section 6.

2 Theoretical Background

This section introduces the existing dual space method for estimating object surface directly from the dual manifold sampled from the silhouettes. Typographically we denote vectors and matrices in bold fonts (e.g., $\mathbf{r}$ and $\mathbf{P}$) and scalar values in italic fonts (e.g., k). A tilde sign over a vector (e.g., $\widetilde{\mathbf{r}}$) represents the homogeneous version of the vector.

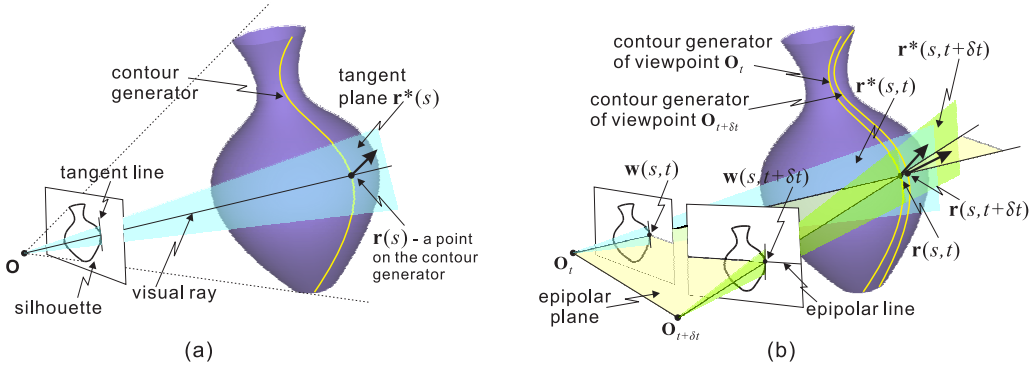

Fig. 1. Geometry of the surface. (a) Single viewpoint; (b) Epipolar parameterization and contour generators

Without loss of generality, we consider a set of pin-hole cameras with known projection matrices $\mathbf{P}_t$, observing an object bounded by a smooth surface S. The camera center, or the viewpoint $\mathbf{O}_t$, is given by ${\mathbf{P}_t}^{\perp}$. A silhouette on the image consists of one or more closed curves delimiting the inside/outside of the object being seen in the image. The back-projection of tangent lines along the silhouette are planes passing through $\mathbf{O}_t$ and tangent to the object surface. The locus of the tangent points on the object surface is the contour generator for the viewpoint $\mathbf{O}_t$ (see Fig. 1(a)). Often the contour generator consists of one or more spatial curves. As the viewpoint moves over time t, it gives rise to a space of tangent planes in $\mathbb{R}^3$. The envelope of this tangent plane space is the original object surface.

Brand et al. [4] showed that the unknown original object surface can be recovered from the tangent plane space sampled from the silhouettes by the virtue of the principle of duality. Under the homogeneous coordinate system, both planes and points are denoted by 4-vectors. Following this track, the tangent plane space of the original surface S can also be treated as a surface in the dual space, denoted as S^*. A surface point $\widetilde{\mathbf{r}}$ and the tangent plane at that point, denoted as $\widetilde{\mathbf{r}}^*$, form a dual pair that satisfies the following symmetric relationship:

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

By the virtue of this symmetry, the duality principle states that we can interchange the roles played by points and planes in many theorems of projective geometry. This leads to the following proposition for computing surface points from tangent planes, which is exactly the dual to the theorem for computing tangent planes from surface points (as for now, we assume that a local para-

metric form for S is available as $S \doteq \widetilde{\mathbf{r}}(s,t)$):

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

where $\widetilde{\mathbf{r}}(s,t)$ is a point on S, $\widetilde{\mathbf{r}}^*(s,t)$ is the tangent plane at $\widetilde{\mathbf{r}}(s,t)$ and also a point on S^*. The 2D equivalence of equation (2) has been formally proved in [4]. The proof for the 3D case, i.e., equation (2), is similar to that of the 2D case, except that the vector cross product for 3-vector is replaced by generalized vector cross product for 4-vector. A geometric interpretation is given in Fig.2.

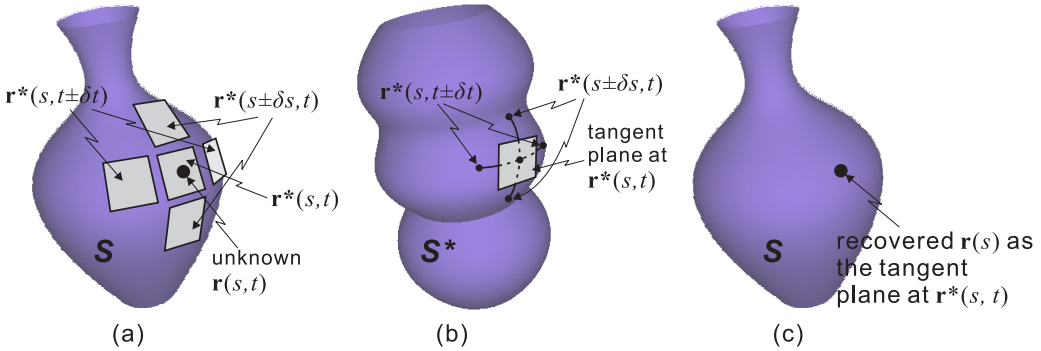

Fig. 2. The dual space theory illustrated. (a) To compute the position of an unknown point $\widetilde{\mathbf{r}}(s,t)$ on the original surface S, the tangent planes at $\widetilde{\mathbf{r}}(s,t)$ and its neighbors are collected. (b) The collected tangent planes are equivalent to points on the dual surface S^*, allowing a tangent plane at $\widetilde{\mathbf{r}}^*(s,t)$ to be estimated. (c) The estimated tangent plane at $\widetilde{\mathbf{r}}^*(s,t)$ on the dual surface is equivalent to $\widetilde{\mathbf{r}}(s,t)$ on the original surface.

In practice, however, the dual surface is sampled from a discrete set of viewpoints, and no well-defined local parametric form for the surface is available. The key challenge involved is therefore how to obtain a reliable tangent basis, in particular $\frac{\partial \widetilde{\mathbf{r}}^*(s,t)}{\partial t}$, in the dual space. Spatial proximity in the dual space was exploited in [4] to qualify neighbor points required for estimating $\widetilde{\mathbf{r}}_t^*(s,t)$. They also sought nearby dual space points only among those sampled from successive views to reduce the risk of picking bad neighbors. However, the reliance on spatial proximity makes it difficult to extend the dual space approach to more complicated shapes in practice. One major reason is that points sharing the same tangent plane (bi-tangent points), as often seen in complicated shapes, are mapped to a single point in the dual space, and the dual surface crosses itself at these singularities. Furthermore, the dual space has a different distance metric than the original space, hence tangent sampled at evenly distributed points in the original space may correspond to ill distributed points in the dual space. As a result, simply using the spatial proximity in the sense of Euclidean distance does not give a correct set of neighbor

points in the dual space.

In our previous work [14], we proposed using epipolar parameterization for obtaining the neighbor information in the dual space. Under this parameterization, the neighbor point along t direction of a contour generator point $\widetilde{\mathbf{r}}(s,t)$, is defined as the intersection between the neighbor contour generator and the epipolar plane (spanned by visual ray of $\widetilde{\mathbf{r}}(s,t)$ and baseline of the two viewpoints). Although the 3D position of this intersection is unknown, its projection on the image can be easily located on the corresponding silhouette. Hence, the dual space point of this point can be computed by back-projecting the tangent line at the projection. The dual space points computed in this way are then used to estimate $\dfrac{\partial \widetilde{\mathbf{r}}^*(s,t)}{\partial t}$, and eventually $\widetilde{\mathbf{r}}(s,t)$, using equation (2).

We illustrate this process in an example given in Fig. 1(b): view $\mathbf{O}_t$ and $\mathbf{O}_{t+\delta t}$ corresponding to two successive images. The epipolar correspondence of $\mathbf{w}(s,t)$, i.e. $\mathbf{w}(s,t+\delta t)$ in the figure, is located by intersecting the epipolar line with the silhouette at time $t+\delta t$. The tangent planes at $\mathbf{w}(s,t)$ and $\mathbf{w}(s,t+\delta t)$, which are $\widetilde{\mathbf{r}}^*(s,t)$ and $\widetilde{\mathbf{r}}^*(s,t+\delta t)$ respectively, are neighbors in the dual space. Once $\widetilde{\mathbf{r}}^*(s,t\pm\delta t)$ is known, $\dfrac{\partial \widetilde{\mathbf{r}}^*(s,t)}{\partial t}$ can be estimated, hence $\widetilde{\mathbf{r}}(s,t)$ with equation (2).

However, a key assumption made in [14] is that the viewpoint undergoes a continuously motion, so the images next to each other in the sequence actually correspond to viewpoints that are spatially adjacent to each other. This implies the contour generators corresponding to successive viewpoints are actually close to each other, and thus the dual space neighbors given by the epipolar parameterization on consecutive views would lead to a good approximation of $\dfrac{\partial \widetilde{\mathbf{r}}^*(s,t)}{\partial t}$. In the next section, we introduce our proposed method to handle more generic situations when viewpoints are randomly distributed and do not follow a specific order.

3 Estimation of Surface Point from Unordered Viewpoints

The assumption about smooth viewpoint motion made by [14] may not be valid in many cases. An example would be a set of viewpoints generated by multiple moving cameras, or a set of viewpoints with no temporal relation. We generalize all these situations as if we are given a set of silhouettes from unordered viewpoints.

Consider a set of n viewpoints with no temporal relation. Accordingly, we drop

the time parameter t and assign to each viewpoint an index i. The contour generator on S corresponding to the viewpoint i is denoted by $\widetilde{\mathbf{r}}_i(s)$, and its dual on S^* by $\widetilde{\mathbf{r}}_i^*(s)$.

The geometric interpretation of the right hand side of equation (2) is the tangent plane at $\widetilde{\mathbf{r}}^*(s,t)$ on the dual surface S^*. Hence, the estimation of surface point $\widetilde{\mathbf{r}}_i(s)$ is equivalent to the estimation of the tangent plane at $\widetilde{\mathbf{r}}_i^*(s)$. To estimate tangent plane at $\widetilde{\mathbf{r}}_i^*(s)$ in a general configuration of unordered views, equation (2) is no longer usable due to the lack of well-defined tangent basis. Alternatively, we approximate the tangent plane from a set of the dual space neighbors of $\widetilde{\mathbf{r}}_i^*(s)$ on S^*, which is very much similar to the estimation of tangent planes on a mesh.

To obtain a set of dual space neighbors, we first need to identify the adjacent viewpoints. The adjacency of viewpoints, however, is not entirely captured by how far they are from each other, but rather by how close their corresponding contour generators are on the object surface. The reason is rather intuitive: we suppose the observed object is bounded tightly by a sphere. The change of contour generator on the surface is more drastic when orbiting the viewpoint around the sphere than moving the viewpoint towards or away from the sphere. An example is given in Fig. 3(a), where d_{12} and d_{13} are the Euclidean distances between the viewpoints $\mathbf{O}_1$ and $\mathbf{O}_2$, and $\mathbf{O}_1$ and $\mathbf{O}_3$ respectively. Although $d_{12} > d_{13}$, the contour generator changes less drastically when switching from viewpoint $\mathbf{O}_1$ to $\mathbf{O}_2$ than from $\mathbf{O}_1$ to $\mathbf{O}_3$. As a result, a more comprehensive measurement is needed for defining the meaning of 'nearby' viewpoints.

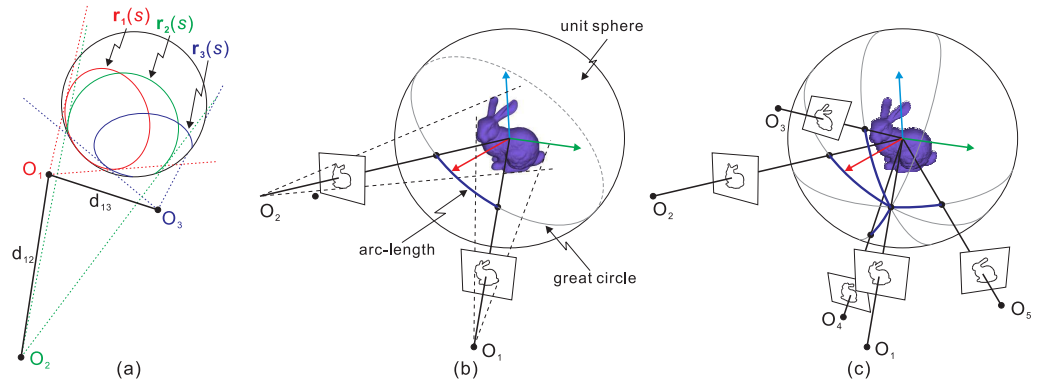

Fig. 3. A general configuration consists of multiple unordered viewpoints. (a) The relative Euclidean distances between the viewpoints do not truly characterize the relative differences between contour generators. (b) The distance between two viewpoints is measured in terms of geodesic arc-length. (c) In the case of more viewpoints, nearby viewpoints are qualified by geodesic arc-lengths.

Let us first consider the case when the viewpoints are relatively far from the object. The ratio between the size of the object and the distance to the viewpoint is very small, and the system is similar to an orthogonal projection. In this case, only orbiting the viewpoint around the object leads to the change of

the contour generator. The adjacency of two contour generators is completely characterized by the directions of their correspondent viewpoints with respect to the object. To measure the directional difference between two viewpoints, it is natural to use arc-length: we conjure a virtual unit sphere at the center of the object and map the viewpoints onto this sphere. The arc-length between two viewpoints i and j is the spherical distance between the mapped viewpoints (see Fig. 3(b)), which is given by:

$$d_{ij} = arccos\left(\frac{(\mathbf{O}_j - \mathbf{C})^\top (\mathbf{O}_i - \mathbf{C})}{|\mathbf{O}_j - \mathbf{C}|\,|\mathbf{O}_i - \mathbf{C}|}\right) \tag{3}$$

where $\mathbf{C}$ is the center of the object. Since we are able to compute the frontier points from the silhouettes [20] which are true feature points on the object, we can approximate $\mathbf{C}$, as well as obtain an approximate size of the object. With the above adjacency measurement, we are able to identify the adjacent viewpoints for any given viewpoint by simply setting a thresholding value θ_0 for the arc-length distance between two viewpoints (see Fig. 3(c)).

Once the adjacent views are identified for each viewpoint, we can proceed to estimate surface points. Each surface point is estimated independently. Let $\widetilde{\mathbf{r}}_i(s_0)$ be the point to be estimated. For each adjacent viewpoint j, we locate the epipolar correspondence of $\widetilde{\mathbf{r}}_i(s_0)$ on the silhouette j, which is the projection of $\widetilde{\mathbf{r}}_j(s_0)$. This gives rise to a tangent plane $\widetilde{\mathbf{r}}^*_j(s_0)$, which is a neighbor point of $\widetilde{\mathbf{r}}^*_i(s_0)$ in the dual space. We can now recast equation (2), which is essentially tangent plane estimation on the dual surface, into a discrete version of weighted tangent plane computation maximally spanning all the neighbor points $\widetilde{\mathbf{r}}^*_j(s_0)$:

$$\widetilde{\mathbf{r}}_i(s_0) = \underset{\mathbf{x}}{\operatorname{argmin}} \sum_{j\in\{j|j\neq i, d_{ij}<\theta_0\}} \left(w_{ji}\frac{(\widetilde{\mathbf{r}}^*_j(s_0) - \widetilde{\mathbf{r}}^*_i(s_0))^\top \mathbf{x}}{|\widetilde{\mathbf{r}}^*_j(s_0) - \widetilde{\mathbf{r}}^*_i(s_0)|}\right)^2 \tag{4}$$

$$w_{ji} = \frac{1}{1+d_{ij}} \tag{5}$$

It should be noted that under perspective projection, moving a viewpoint towards or away from the object will also cause subtle change to the contour generator. We can characterize the extent of this change using the surface normal at the contour generator points. As shown in Fig.4, moving the viewpoint orbitally has a linear relationship to the change of the normal. Changing the distance to the object, on the other hand, does not result in a linear change of the normal. The extent of the change also depends on the normal of the tangent plane, as well as the size of the object.

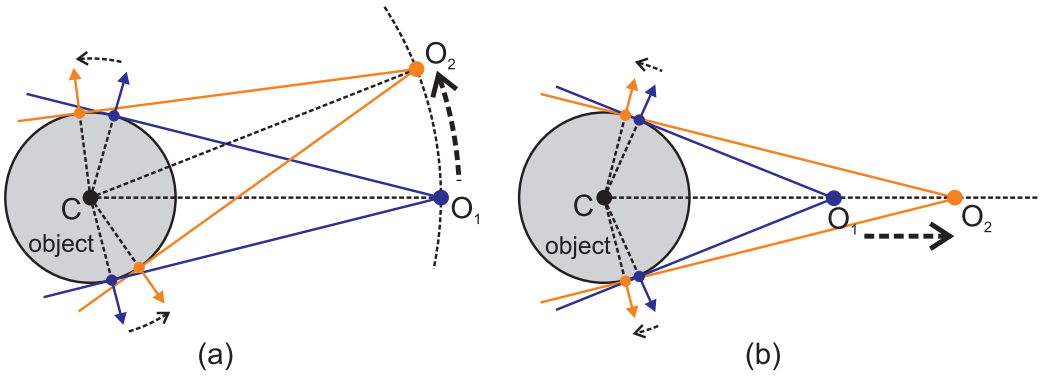

Fig. 4. Change of contour generator on a sphere due to different type of viewpoint motion under perspective projection. (a) Viewpoint moves orbitally, and the change of the contour generator is linear to the angle $\mathbf{O}_1\mathbf{C}\mathbf{O}_2$. (b) Viewpoint moves towards or away from $\mathbf{C}$, the change of the contour generator is not linear with respect to the change of distance to $\mathbf{C}$.

To account for the effect brought by non-orbital viewpoint motion under perspective projection, we tune the weighting w_{ji} in equation (5) by introducing a term that penalizes such viewpoint motion:

$$w_{ji} = \frac{1}{1 + \lambda_1 d_{ij} + \lambda_2 \left| arccos(\frac{h}{|\mathbf{O}_j - \mathbf{C}|}) - arccos(\frac{h}{|\mathbf{O}_i - \mathbf{C}|}) \right|} \tag{6}$$

where λ_1 and λ_2 are constants, h is half of the approximated diameter of the object. We choose to use $arccos(\frac{h}{|\mathbf{O}_j - \mathbf{C}|})$ here, because it reflects the extent of the change of the tangent plane to the object's bounding sphere with respect to the change of the distance between the viewpoint and the object. In practice, we can safely assume that $h < |\mathbf{O}_i - \mathbf{C}|$, because the viewpoints are always placed outside the bounding volume of the object.

To summarize, our proposed algorithm first identifies a set of adjacent viewpoints for each viewpoint using arc-length. To estimate a point on a contour generator, we collect the tangent plane at the "epipolar correspondence" of this point from each adjacent viewpoint. The collected tangent planes, or equivalently, the dual space points, are used to compute the position of the contour generator point we wanted to estimate, by applying equation (4) with proper weightings described in equation (6). In this way, we recover all the contour generators, together with the surface normal at each point (directly inferred from the tangent plane at that point). The algorithm is summarized in Algorithm 1.

Algorithm 1 Estimation of Surface Points

Initialize viewpoint adjacency matrix d and weighting matrix w
Approximate object center $\mathbf{C}$ and diameter h
for i^{th} viewpoint **do**
  for j^{th} viewpoint $(j > i)$ **do**
    compute $d(i, j)$ using eqt. (3)
    compute $w(i, j)$ using eqt. (6)
  end for
end for
Initialize 2D storage $\widetilde{\mathbf{r}}(s, t)$ for storing surface points
for i^{th} silhouette $\mathbf{w}_i(s)$ **do**
  for n^{th} point $\mathbf{w}_i(s_n)$ on the silhouette **do**
    for j^{th} viewpoint $d(i, j) < \theta_0$ **do**
      locate epipolar match $\mathbf{w}_j(s_n)$
      compute $\widetilde{\mathbf{r}}_j^*(s_n)$
    end for
    Estimate surface point $\widetilde{\mathbf{r}}_i(s_n)$ using $\left\{\widetilde{\mathbf{r}}_j^*(s_n)\right\}$ and $w(i, j)$
  end for
end for

4 Extraction of Surface Mesh

Our next goal is to extract a surface mesh from the contour generators computed in the previous section. Each viewpoint produces a contour generator in the form of one or several spatial curves. The spatial curves produced in this way from all viewpoints form a web on the original object. The number of spatial curves produced by each viewpoint depends on two factors: the topology (or formally, the aspect) of the corresponding silhouette and self-occlusions. Self-occlusions on the surface may lead to wrong epipolar matching, making the tangent estimation mentioned in Section 3 unreliable. Although it usually accounts for only a small portion of all the points estimated, it may still lead to discontinuities along the recovered contour generators. Besides, as we make no assumption on the topology of the object surface, it is a great challenge to directly triangulate the recovered surface points while conforming to the topology suggested by the silhouettes.

Boyer and Franco [3] proposed to apply directly the 3D Delaunay triangulation and refine the topology of the resulting triangulation by carving away those tetrahedrons projecting outside any silhouette. However, setting aside the difficulty of 3D Delaunay triangulation (as compared with the 2D version), the resulting triangulation may be over-carved if the sampling density along each silhouette is not carefully chosen.

In our approach, we keep the idea of carving for its robustness, but on 2D

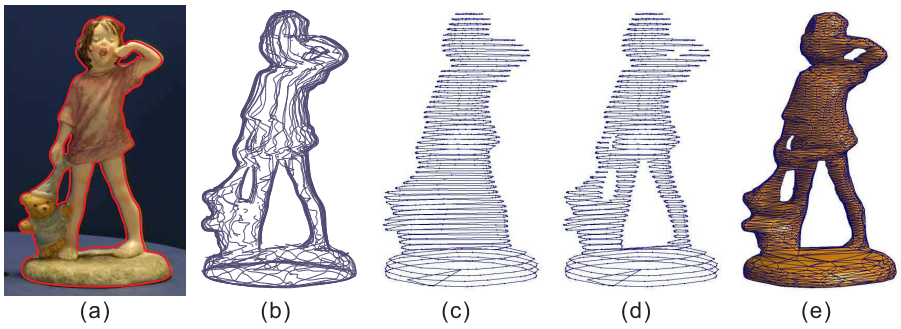

Fig. 5. From contour generators to surface mesh. (a) One of the input silhouettes from "Girl and Teddy Bear" sequence. (b) The contour generators computed using the process introduced in Section 3. (c) 2D polygons on the slicing planes before being pruned by the input silhouettes. (d) 2D polygons on the slicing planes after being pruned by the input silhouettes. (e) The resulting surface mesh, containing 4 779 triangles.

Delaunay triangulation instead of 3D. This approach is motivated by a robust algorithm for surface extraction from cross-sectional contours of arbitrary topology [1] and one of its implementation - NUAGES.

Firstly, our problem is reduced from 3D to 2D: we resample the web of contour generators by parallel slicing planes, similar to what have been done in [14]. Vaguely, the resampling process is to use parallel slicing planes to cut the web of contour generators. This results in a number of isolated points (the intersection between the contour generators and the slicing plane) on each of the 2D slicing planes. Now for each slicing plane, we triangulate the isolated points with 2D Delaunay triangulation. Each node (i.e., each triangle) in the 2D Delaunay triangulation is validated against all silhouettes, and those projecting outside any silhouette are marked for deletion. We repeat this process for all slicing planes and finally obtain a stack of triangulations on the slicing planes all conforming to the topology suggested by the silhouettes (see the example given in Fig. 5).

To obtain the object surface, we transfer the 2D Delaunay triangulation of every two adjacent slicing planes to 3D tetrahedrons. This process has been introduced in [1] and we will omit the details. An example is given in Fig.6. Briefly speaking, each node on either slicing plane gives birth to a type 1 tetrahedron (with a facet on either slicing plane), and a pair of intersecting Voronoi edges (one from each slicing plane) produces a type 2 tetrahedron (without any facet on either slicing plane) which is joint with type 1 tetrahedrons sharing a facet. Instead of going through the post-processing step as suggested in [1], we mark all type 1 tetrahedrons for deletion if it is constructed from a node on either slicing plane which is marked for deletion previously. Finally, the object surface can be extracted as the visible facets of both types of tetrahedrons. "Visible" here means a facet on an unmarked tetrahedrons and satisfying: 1) it is not shared with another unmarked tetrahedron, or 2) it is shared with

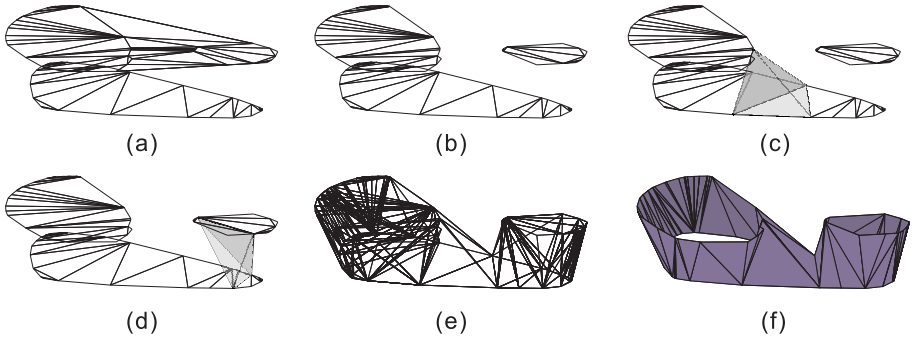

Fig. 6. Extracting surface from two adjacent slicing planes. (a) 2D Delaunay triangulation. (b) 2D Delaunay triangulation with its topology corrected. (c) An example of type 1 tetrahedron. (d) An example of type 2 tetrahedron. (e) All the type 1 and 2 tetrahedrons constructed. (f) The surface extracted from "visible" tetrahedrons.

another tetrahedron which has been marked for deletion. A detailed outline of the algorithm is given in Algorithm 2.

Algorithm 2 Extracting the object surface mesh

```
resample the contour generators by parallel slicing planes
for each slicing plane do
    form 2D Delaunay triangulation for this slicing plane
    for each node in the 2D Delaunay triangulation do
        mark the node for deletion if it projects outside any silhouette
    end for
    for each node in this and the previous slicing plane do
        form type 1 tetrahedrons
        if this node is marked for deletion then
            mark the tetrahedron for deletion
        end if
    end for
    form type 2 tetrahedrons and connect with neighbor type 1 ones
    mark all non-solid[1] tetrahedrons for deletion
    for each type 1 and type 2 unmarked tetrahedrons do
        for each face of the tetrahedron do
            if the face is "visible" then
                store the face as the part of final surface mesh
            end if
        end for
    end for
end for
```

The example given in Fig. 5 shows the intermediate steps of the surface extracting process and the final surface mesh. The input is a sequence of silhou-

[1] Solid tetrahedrons are those sharing a facet with a type 1 tetrahedron or another solid tetrahedron not marked for deletion. A type 1 tetrahedron is solid by itself.

ettes from 20 calibrated viewpoints. From the experiments in Section 5, we can see that the proposed surface extraction algorithm robustly handles rather complicated object surface. As the topology correction and surface extraction are independently done between two adjacent slices, the overall computational time increases only linearly with the number of slicing planes.

Compared with [3], the 2D Delaunay triangulation is easier to construct and yields better-shaped tetrahedrons (due to the resampling), hence better-shaped mesh triangles. Compared with [14], we take advantage of 2D Delaunay triangulation and carving for determine the topology, which in theory is robust to any topology.

A note to be taken is that in order to reconstruct a relative completely and topologically correct model, the proposed surface extraction method requires a reasonably sufficient number of contour generators to be recovered. Too few recovered contour generators may lead to very large tetrahedrons that lie partially outside of the visual hull, and hence can potentially be marked for deletion mistakenly. This problem can be remedied by subdividing the large tetrahedrons and repeating the projection test for the subdivided tetrahedrons. However, this is usually not necessary in practice, because the surface point estimation algorithm we have adopted is essentially a differential approach, which requires a reasonably dense and well-distributed image sequence in the first place.

5 Experiments

In this section, we show both the quantitative and qualitative results of our approach tested with synthetic data and real world image sequences.

5.1 Synthetic Data

The synthetic experiments are used to evaluate the accuracy of surface point estimation and the stability of surface extraction algorithm. In [14], it has been demonstrated that using a dual space approach gives more accurate surface point estimation than octree and EPVH. The work has given detailed quantitatively analysis on this subject as well. The quantitative result here is an extension to the analysis of [14], which demonstrates that the dual space method extended non-linear sequences gives equally accurate, if not better, estimation of the contour generator points.

A set of 3D models (see Fig. 7) with varying complexity and topology are

used, including venus, vase (surface of revolution), torus (surface with non-zero genus), bunny (surface with fairly complicated shape), and knot (surface with very complicated topology). We compute the error of reconstruction as the average point-to-surface distance with respect to the ground truth surface. We normalize the error against the size of the object, which is the diameter of the minimum bounding sphere. Hence, 1% of error represents an average distance of 0.01 if the object size is 1.

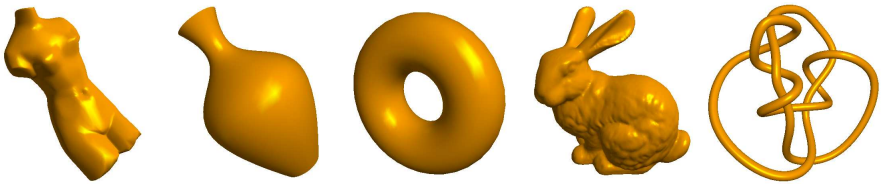

Fig. 7. The set of synthetic models used for the evaluation of reconstruction error. From left to right: venus, vase, torus, bunny and knot.

We compare the results of our approach with a standard octree based volumetric approach used in [18,20], as well as EPVH [1] [8], which aims to produce exact polyhedral representation of the visual hull. To be fair, the three algorithms are tuned to generate similar number of triangles.

As shown in Table 1, EPVH improves over octree for not relying on spatial discretization. The resulting estimations are guaranteed to lie on the visual ray touching the object surface. A limitation here is that these points, however carefully computed, are only guaranteed to lie on the visual hull encapsulating the real surface. It has already been shown in [14] that the dual space approach gives more accurate estimation than the octree and EPVH approaches. The approach in this paper has addressed a main drawback of [14] to handle sequences of unordered viewpoints. Conveniently, our estimation is no longer bound to using two adjacent viewpoints along the camera motion path, but rather all adjacent viewpoints identified. Hence it comes as no surprise that our approach is more accurate in estimating the real surface points than both octree and EPVH.

Recon. Err.	Knot95	Venus	Bunny	Torus	Vase
Octree	0.525%	1.532%	0.949%	3.671%	1.153%
EPVH	0.379%	0.831%	0.498%	3.275%	0.652%
Our Approch	0.302%	0.729%	0.430%	2.341%	0.130%

Table 1
Comparison of the reconstruction error between our approach, octree approach and EPVH approach. 1% error here is equivalent to 1 centimeter if the minimum bounding sphere has the diameter of 1 meter.

[1] The implementation of the EPVH is obtained from http://perception.inrialpes.fr/ Franco/EPVH/

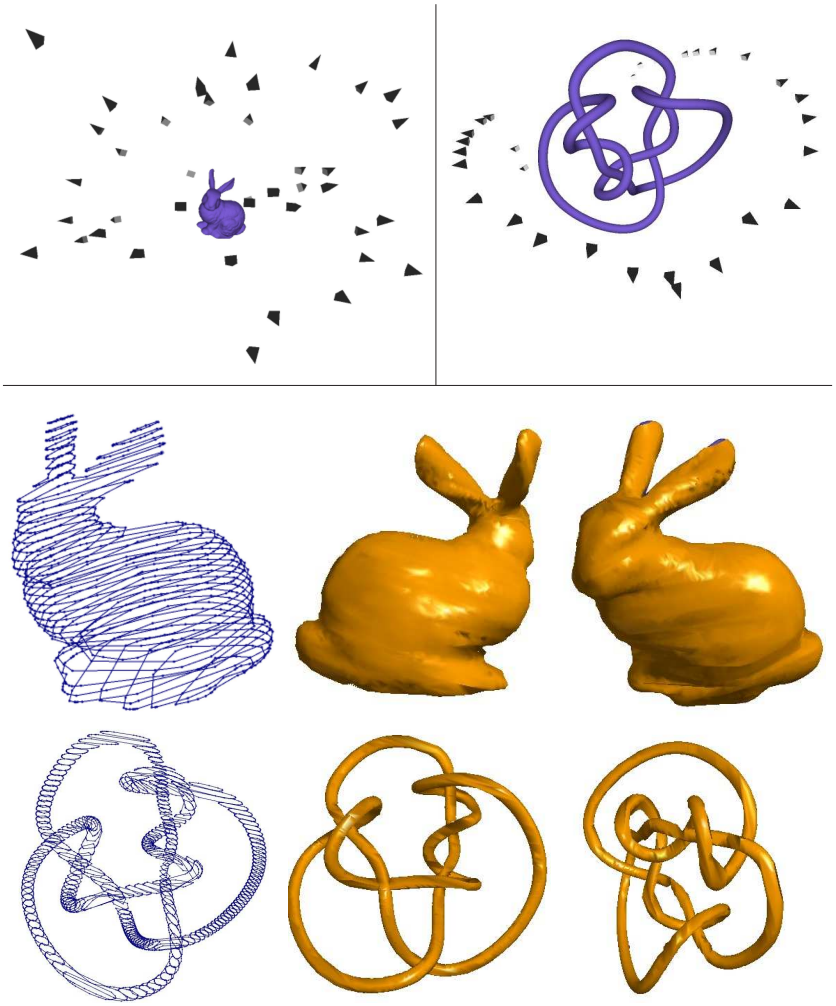

Fig. 8. "Stanford Bunny" sequence and "Knot 9_5"[1] sequence. (top) The viewpoints used for the bunny sequence (39 in total) and for the knot sequence (34 in total). (middle) The polygons on slicing planes[2] and the reconstructed mesh for the bunny sequence. (bottom) The polygons on slicing planes[2] and the reconstructed mesh for the knot sequence.

Two example reconstruction results of our proposed approach are shown in Fig. 8. On top, the viewpoints of the input image sequences are shown. The bunny sequence shows that our algorithm works with a non-linear set of random viewpoints, provided that they are reasonable well distributed around the object. The viewpoints for the knot sequence, given the fact that they are more regularly distributed in a ring around the object, are shuffled before input into our algorithm to eliminate any implicit order information. The accuracy of the reconstructions of these two objects has been demonstrated in Table 1. The middle and bottom of Fig. 8 have also shown that the topology of the original object is also retained.

[1] The original model can be found at http://www.pims.math.ca/knotplot/zoo/
[2] Only half of all slices are shown for clarity of display

5.2 Real World Data

Our proposed algorithm has also been tested with real world image sequences (see Fig. 9). The input images are calibrated and their order is scrambled to ensure that no temporal order information of the images is available. The silhouettes of the object in the images are extracted using active B-Spline [6], which allows tangent planes on the silhouettes to be computed conveniently, as well as the intersection between epipolar lines and the silhouettes to be computed with a closed form solution.

Fig. 9. A few original images for the "Hannover Dinosaur" sequence, the "Girl and Teddy Bear" sequence, and the "Porcelain Lady" sequence.

The "Hannover Dinosaur" sequence (courtesy of University of Hannover) consists of 36 images. The results are shown in Fig. 10 for octree, EPVH and our approach. To extract the surface mesh from the octree, mid-point [16] interpolation is adopted due to the binary nature of the occupancy information embedded in the octree. We did not impose any additional smoothing to the resulting mesh, which trades high frequency information for perceived visual appeal. We also tuned all approach to produce around 30 000 triangles in this example. The octree approach, without mesh smoothing, generated very bumpy surface. Compared with octree, both EPVH and our approach retain good shape details of the original object, such as the shape of spine. This is expected because of the way they honor the information provided by each silhouettes. However the mesh resulting from EPVH is inheritably jagged, due to the irregularity of the shape and size of the triangles formed from joining the visual ray segments. In addition, the complicacy involved in joining these segments could lead to cracks like the one shown in Fig. 10(c), near the inner side of the tail of the dinosaur model. Our proposed method is based on well established delaunay triangulation to guarantee the water-tightness on each slicing plane. Although the slicing planes adopted are almost parallel to the arms of the dinosaur model and the initial 2D triangulation can be very wrong on the slicing planes, the volumetric nature of pruning the unwanted volume trunks, in this case, the delaunay cells, results in topologically correct mesh.

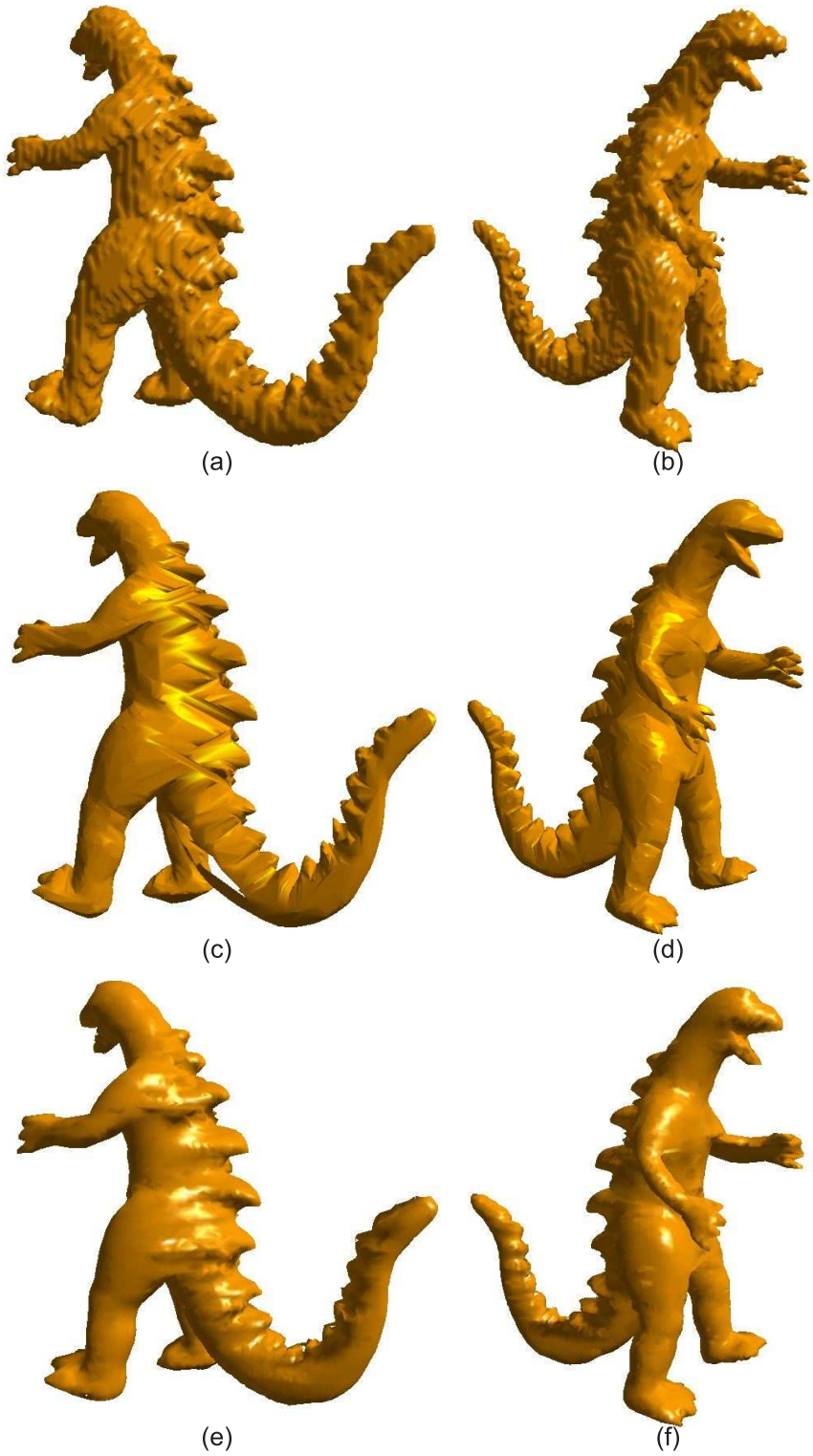

Fig. 10. The "Hannover Dinosaur" sequence. (a-b) Two novel views of the surface mesh produced by octree. (c-d) Two novel views of the surface mesh produced by EPVH. (e-f) Two novel views of the surface mesh produced by our proposed method.

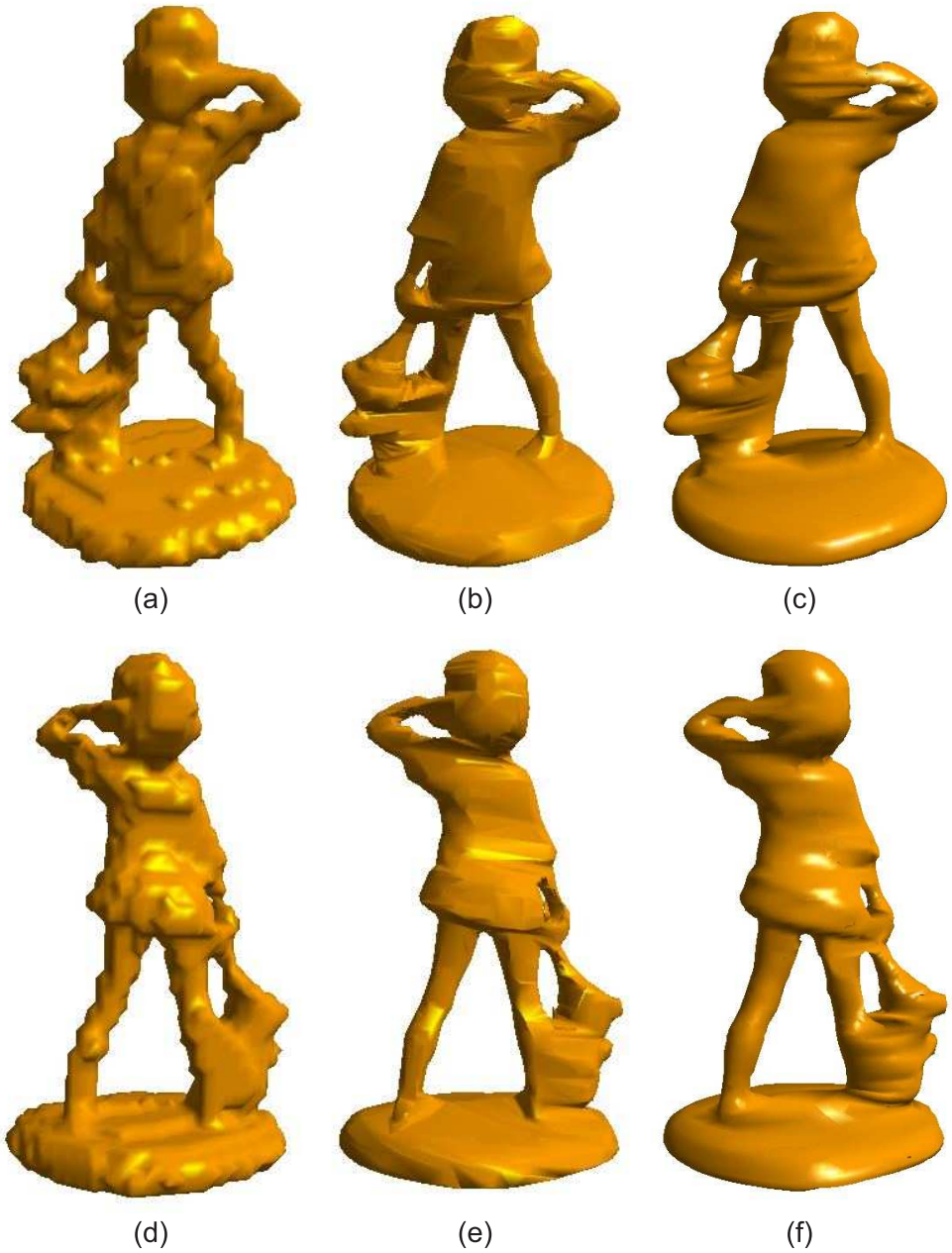

Fig. 11. The results from "Girl and Teddy" sequence. (a)(d) Two novel views of the surface mesh produced by octree. (b)(e) Two novel views of the surface mesh produced by EPVH. (c)(f) Two novel views of the surface mesh produced by our proposed method.

Two more real world sequences are shown below, including a "Girl and Teddy Bear" sequence consisting of 20 images (see Fig. 11), and a "Porcelain Lady" sequence consisting of 36 images (see Fig. 12, courtesy of University of Cambridge). The reconstructed mesh for the former contains around 8 000 triangles, and the later 10 000 triangles. Similar to the dinosaur sequence, the meshes produced by octree bear clearly the sign of spatial discretization. Those

by EPVH and our approach, with almost the same amount of triangles, are able to capture the shape observed from silhouettes very well. However, the visual hull mesh, no matter how accurate, does not represent the true object surface. Furthermore, the triangles formed by joining visual ray segments have a great variance on shape and size. These explain why in the EPVH's result shown above, the shading is jagged on the surface areas which are supposed to be smooth, such as the back of the girl in Fig. 11(e), and the lower part of the robe in Fig. 12(b). On the other hand, our approach produces a mesh with vertices consisting of hypothetic real surface points, and as a bonus, the true surface normal at these vertices as well. In this way, the local surface properties are much better preserved. Reflected in the results, the surface meshes produced by our approach are free of the artifacts appearing on polyhedral visual hull meshes.

6 Conclusion and Discussion

We have introduced a novel differential method for reconstructing surface from silhouettes. The proposed method is capable of producing highly accurate estimation of the surface points, yet requires no prior knowledge concerning the order of the viewpoints. A surface extraction algorithm is also proposed to deal with objects with unknown topology. Unlike many other differential approaches, our approach is robust to object with non-zero genus and complicated topology as most of volumetric approach. We verified our approach with both synthetic and real world data, and we also compare the result with existing widely used methods.

The 3D reconstruction produced by the current method, although complies to the silhouette constraint, does not reflect the concavities on the original object surface. To tackle this problem, photo-consistency or shading information must be incorporated. Since we do have the exact position of contour generators, local refinement becomes possible, which leads to possible further extension of the current work.

7 Acknowledgement

This project is supported by a grant from the Research Grants Council of the Hong Kong Special Administration Region, China, under Project HKU 7180/06E.

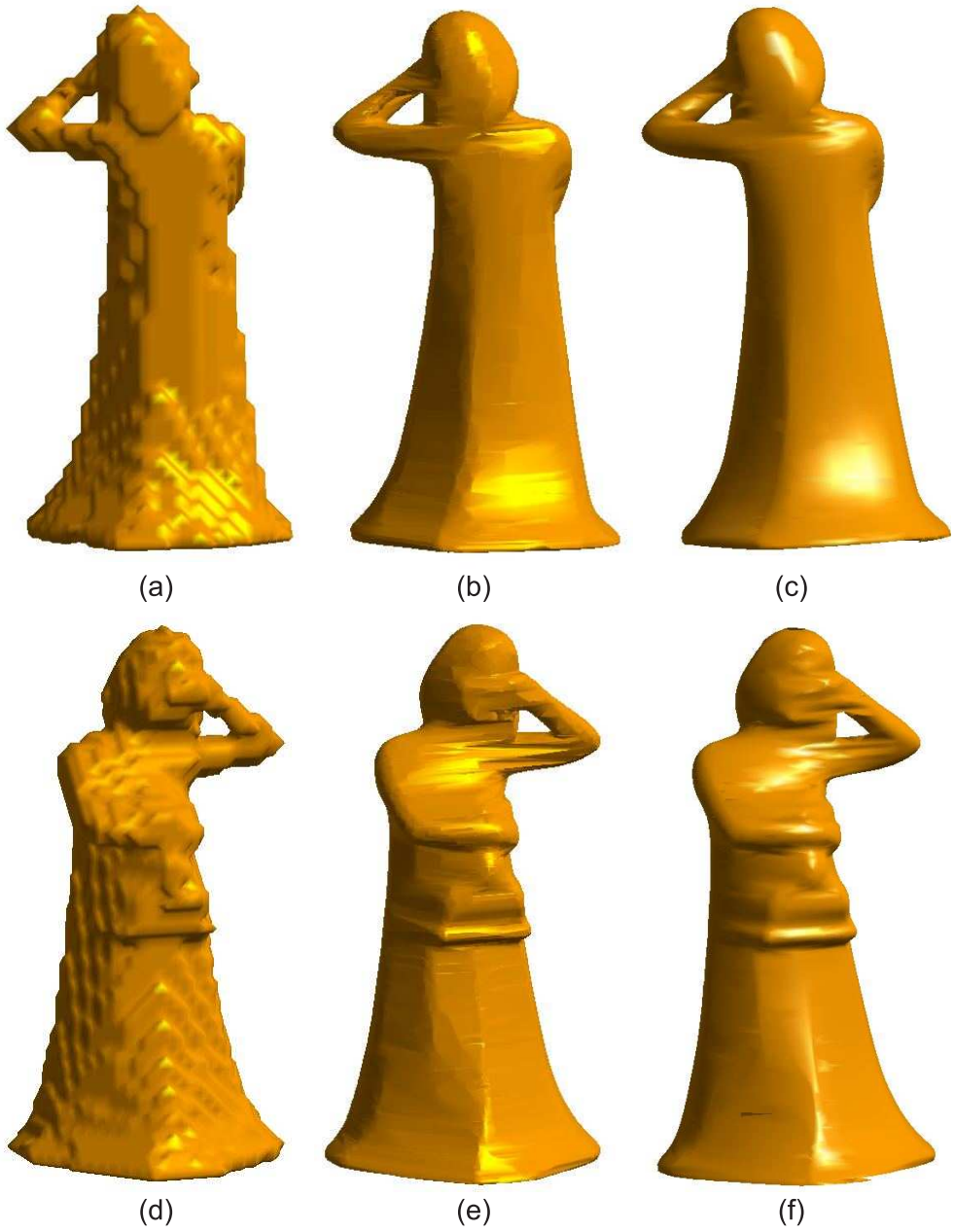

Fig. 12. The results from "Porcelain" sequence. (a)(d) Two novel views of the surface mesh produced by octree. (b)(e) Two novel views of the surface mesh produced by EPVH. (c)(f) Two novel views of the surface mesh produced by our proposed method.

References

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

[2] E. Boyer and M.-O. Berger. 3d surface reconstruction using occluding contours.

In *Computer Analysis of Images and Patterns*, pages 198–205, 1995.

[3] 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, USA, June 2003.

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

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

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

[7] G. Cross and A. Zisserman. Surface reconstruction from multiple views using apparent contours and surface texture. In *NATO Advanced Research Workshop on Conference of Computer Vision and Computer Graphics*, pages 25–47, 2000.

[8] J.-S. Franco and E. Boyer. Exact polyhedral visual hulls. In *British Machine Vision Conference*, volume 1, pages 329–338, September 2003.

[9] B. García and B.G. Brunet. 3D reconstruction with projective octrees and epipolar geometry. In *International Conference on Computer Vision*, pages 1067–1072, January 1998.

[10] P.J. Giblin and R.S. Weiss. Reconstruction of surfaces from profiles. In *International Conference on Computer Vision*, pages 136–144, London, England, June 1987.

[11] C. Hernández and F. Schmitt. Silhouette and stereo fusion for 3d object modeling. *Computer Vision and Image Understanding.*, 96(3):367–392, 2004.

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

[13] S. Lazebnik. Projective visual hulls. Master's thesis, University of Illinois at Urbana-Champaign, 2002.

[14] C. Liang and K.Y.-K. Wong. Robust recovery of shapes with unknown topology from the dual space. In *Pattern Analysis and Machine Intelligence (PAMI)*, volume 29, pages 2205–2216, 2007.

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

[16] C. Montani, R. Scateni, and R. Scopigno. Discretized marching cubes. In *IEEE Conference on Visualization*, pages 281–287, Washington, DC, October 1994.

[17] M. Potmesil. Generating octree models of 3d objects from their silhouettes in a sequence of images. *Computer Vision, Graphics and Image Processing*, 40(1):1–29, October 1987.

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

[19] R. Vaillant. Using occluding contours for 3D object modeling. In *European Conference on Computer Vision*, pages 454–464, Antibes, France, April 1990.

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