Extracting Surface Representations from Rim Curves

Hai Chen[1], Kwan-Yee K. Wong[2], Chen Liang[2], and Yue Chen[1]

[1] College of Software Technology, Zhejiang University, Hangzhou, Zhejiang, China 310027
chenhai@gmail.com
chenyue@cs.zju.edu.cn

[2] Department of Computer Science, The University of Hong Kong, Pokfulam Road, Hong Kong
{kykwong, cliang}@cs.hku.hk

Abstract. In this paper, we design and implement a novel method for constructing a mixed triangle/quadrangle mesh from the 3D space curves (rims) estimated from the profiles of an object in an image sequence without knowing the original 3D topology of the object. To this aim, a contour data structure for representing visual hull, which is different from that for CT/MRI, is introduced. In this paper, we (1) solve the "branching structure" problem by introducing some additional "directed edge", and (2) extract a triangle/quadrangle closed mesh from the contour structure with an algorithm based on dynamic programming. Both theoretical demonstration and real world results show that our proposed method has sufficient robustness with respect to the complex topology of the object, and the extracted mesh is of high quality.

1 Introduction

In 3D model reconstruction from image sequences, silhouettes are often a reliable and obvious feature that can be extracted from the images easily. With the knowledge of the epipolar geometry, which governs the relative positions and orientations between cameras, it is possible to recover the 3D space curves lying on the surface of the object that are projected onto the images as the silhouettes. In the literature, such 3D space curves are known as contour generators or rims.

Various techniques[1, 2, 3] have been developed for estimating rims from silhouettes. The rims so obtained carry not only 3D positional information of the space curves, but also surface information like the surface normal. It is desirable to extract a surface representation of the object from the rims. However, this is not a trivial task as the rims cannot be recovered perfectly. Very often, the rims recovered may be discontinuous due to self-occlusion. Besides, they may intersect with each other at frontier points (see [1] for details). The points forming the rims are also not evenly spaced. Hence, direct triangulation of the points on the rims using existing algorithms often cannot produce satisfactory results.

In this paper, a novel method for constructing a surface mesh from the rims estimated from the silhouettes is introduced. Instead of triangulating the rims points directly, the rims are first re-sampled by evenly spaced parallel slicing planes. The sample

P.J. Narayanan et al. (Eds.): ACCV 2006, LNCS 3852, pp. 732–741, 2006.

points on each slicing plane then forms the 2D cross-section contours of the object. The problem of forming a surface mesh is then converted into the problem of joining these cross-section contours on adjacent layers. Unlike cross-section data commonly seen in MRI/CT related research, we observe that the contours recovered from silhouette data do not always overlap with each other. This is the well-known problem of ”branching structure” in MRI/CT visualization. A method based on “directed edge” is introduced here to solve this problem. By exploiting the dynamic programming (DP) techniques, it is shown that the mesh ‘belt’ can be reconstructed from two adjacent cross-section contours with a low computation complexity, which is linear to the product of the numbers of the mesh vertices on the two contours. This technique also allows us to build a high quality mesh in terms of surface smoothness.

Another contribution of this paper is that the final surface can be presented in the form of a mixed triangle/quadrangle mesh, which can be rendered more efficiently than the pure triangle mesh. The mixed triangle/quadrangle mesh has only been used in mesh subdivision and mesh edit, and it is the first time that it is extracted directly from image data.

The latter of this section gives the overview of related works. Section 2 presents the theoretical background of mesh. Section 3 describes the algorithms and implementations for extracting a mesh surface from the rims. Experimental results from real models will be given in Section 4, and Section 5 concludes this paper.

1.1 Related Works

In the literature, there have been some related researches that attempt to extract a mesh from the rims of an object. In [2], the triangular mesh is extracted directly based on the relationship between neighboring rims with minimal computation complexity. However, such a relationship between rims will not hold if the rims are fragmentary, and this happens quite often for complex shapes with non-zero genus. Note that this approach makes no guarantee on the quality nor the obturation of the outcome mesh.

Since the connectivity information is implied for points recovered along the rims, such points can be reformed into another data structure which makes the surface extraction easier. In this paper, the rims are re-sampled into cross-section contours. There are numerous researches on the contour data, but most of them are based on contour data derived from CT/MRI. In [4], Cong and Parvin recovered a surface from planar sectional contours based on the “Equal Importance Criterion” which suggests that every point in the region contributes equally to the reconstruction process. This algorithm derives the iso-surface constructed by PDE and the primitive representations by Voronoi Diagram transformation. However, this approach is of very high complexity and not suitable for the surface reconstruction of the visual hull.

There are also attempts made to extract a surface from contours for visual hull reconstruction. In [5], Boissonnat exploited Delaunay’s triangulation, a method commonly used in reconstructing 3D surface from unorganized points, to extract a surface from the planar contour structure. The algorithm is robust but the resulting surface is not obturated. Besides, the algorithm produces low quality mesh near branching structures on the surface.

2 Theoretical Background

2.1 Quality of Mesh Elements

A mesh with high quality not only can faithfully capture the true topology of a complex 3D object, but also can be rendered efficiently. According to [6], a high quality visual hull surface should be a compact, connected, orientable, two dimensional manifold, and with or without boundary.

Let us first define quantitatively a measurement of the quality for the mesh elements. Traditionally, the quality of a mesh triangle is measured by the smallest internal angle, and the quality of the triangle is said to increase with its smallest internal angle [7]. Since in this paper, our output mesh will be a mixed triangle/quadrangle mesh (to be introduced in next subsection), the internal angle measurement cannot be applied. Here, a distance measurement similar to that used in [8] and [9] is used to measure the quality of triangles and quadrangles.

During the extraction from cross-section contours, every mesh elements are formed from points on the contours lying on adjacent layers, and are either a triangle or a quadrangle. A triangle consists of a vertex from the contour polygon on one layer and an edge of a contour polygon on another layer. By projecting the vertex onto the other contour plane, the ***Error of Triangle*** is defined as the squared distance between the projection (p_i) and the center (c_j) of the edge (see (1) and (2)).

A quadrangle consists of an edge of the contour polygons lying on one layer and an edge of the contour polygons lying on the other layer. By projecting the center of one edge onto the other contour plane, the ***Error of Quadrangle*** is the defined as twice the squared distance between the projection (c_i) and the center (c_j) the other edge (see (3)).

2.2 Mixed Triangle/Quadrangle Mesh

A mesh formed using the triangle scheme can retain sharp features more faithfully, while that formed using a quadrangle scheme is more suitable for representing smooth surfaces. Triangle meshes generate poor limiting surface when using quadrangle-only scheme, while quadrangle meshes behave poorly with triangle-only scheme. To increase flexibility, both triangle and quadrangle schemes are needed in modeling real world data. Recently, Stam and Loop [9] introduced a new subdivision operator that unifies mixed triangular and quadrilateral subdivision schemes on C^1 surfaces. Latter, Schaefer and Warren [8] proved that mixed triangle/quadrangle scheme mesh could be used in C^2 surfaces. Here, in this paper, the mixed triangle/quadrangle meshes on C^2 surfaces are extracted from the contour data structure directly.

3 Surface Extraction from Rims

In this work, the cross-section contours are first formed from the rim fragments estimated from the silhouettes. A dynamic programming based method is then introduced to produce a high quality triangle/quadrangle mesh from these cross-section contours. We will explain the algorithm and our implementation in detail.

3.1 Contour Data from Rims

To extracting a high quality mesh, the rims are first transformed into a more efficient data structure bearing the topological information observed from the silhouettes. A cross-section contour data structure is adopted here for representing the surface of the visual hull.

We adopt the method introduced in [1] to recover the rims from the silhouettes in an image sequence. These rims are then re-sampled into cross-section contours by parallel slicing planes. The normal of the slicing planes are chosen to be the direction parallel to the longest shaft of the original object. To recover the contour structure for complex models, we back-project the points onto the extracted silhouettes and regroup points into one or more contours on the same sliced plane. After regrouping, points on each cross-section form one or several planar polygons (contour polygons) which correctly capture the topology observed from the silhouettes.

As mentioned earlier, the recovered rims are inevitably fragmentary. Moreover, the number of rim curves are limited by the number of images/cameras. As a result, the edge of the contour polygons at places where the rims are very sparse will be very long. During reconstruction, long edges will lead to ill-formed triangles. In this paper, long edges are subdivided by inserting additional points along it. A more aggressive scheme is also possible: since we know the surface normal for each vertex of the contour polygon, long edges can be replaced with fitted parabola curves to make it look smoother. Since the surface normal at each vertex is known, we can make sure the contour polygons, and hence the final surface, always fall within the visual hull defined by the silhouettes, while making them look smoother and aesthetically pleasing. Extracting Surface Representations From Rim Curves.

3.2 Mesh Extraction from Cross-Section Contour

The major difficulty in the extraction of a mesh from cross-section contours is the branching problem [5]. Here, one reasonable assumption is made that the object is not extremely skew and the intercrossing planes are dense enough to present the topology.

Let us consider two adjacent contours, and denotes $\mathcal{A}_i$ the contour polygon on one layer and $\mathcal{W}_j$ the contour polygon on the layer immediately below.

Definition 1. *If and only if the center of $\mathcal{A}_i$ can be projected within $\mathcal{W}_j$ or the center of $\mathcal{W}_j$ can be projected within $\mathcal{A}_i$, $\mathcal{A}_i$ and $\mathcal{W}_j$ have a* ***connectedness relationship****.*

Definition 2. $\langle \{\mathcal{A}_1, \ldots, \mathcal{A}_m\}, \{\mathcal{W}_1, \ldots, \mathcal{W}_n\} \rangle$ *is an* ***m-n connectedness pair*** *if and only if:*

- *Neither* m *nor* n *is 0,*
- *For any $\mathcal{A}_i \in \{\mathcal{A}_1, \ldots, \mathcal{A}_m\}$, there exists $\mathcal{W}_j \in \{\mathcal{W}_1, \ldots, \mathcal{W}_n\}$ that $\mathcal{A}_i$ and $\mathcal{W}_j$ are having a connectedness; For any $\mathcal{W}_k$ having a connectedness with any $\mathcal{A}_i \in \{\mathcal{A}_0, \ldots, \mathcal{A}_m\}$, $\mathcal{W}_k \in \{\mathcal{W}_0, \ldots, \mathcal{W}_n\}$ is true and vice versa.*

According to the definition above, the *1-1 connectedness pair* corresponds to **a simple structure**, while *others* are **branching structure** [10, 11] (see Fig. 1(a) and Fig. 2(a)).

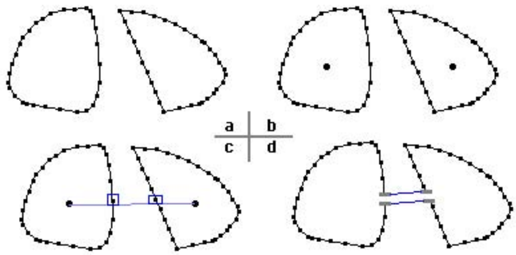

Fig. 1. (a) Contour data with branching structure. (b) Centers of the contour polygons. (c) Vertices to be slipped is selected by considering the line joining the two centers. (d) Conversion of the structure into a simple structure with the additional directed edge pair.

Branching structure is a challenging problem in contour reconstruction. The major difficulty is how to slip from one branch to another automatically. Further, the output mesh should be obturated.

To handle the branching structure, an additional directed edge pair is introduced. The additional directed edge pair is formed by two edges with the same ends but opposite directions. They are used to decide where to slip from one polygon into another. The aim is to emerge all the polygons on each side of the m-n connectedness pair. Here, we introduce a method to gain the additional directed edge pair.

First, consider a 2-1 branching structure (see Fig. 1). The centers of the two polygons that to be connected are first computed (see Fig. 1(b)). Two vertices each from one polygon are then selected. The vertex should be the one closest to the line segment connecting the centers of the two polygons (see Fig. 1(c)). Finally, a directed edge pair with these selected vertices as end points is added between the two polygons, and the branching structure is transformed into a simple structure (ss Fig. 1(d)).

Actually, the real world topology might be much more complex than a 2-1 branching structure. We solve it by recursively applying the above method. To do this, each time we pick two polygons on one slicing plane having the shortest center distance. These polygons form a 2-1 branching structure and can be solved by earlier mentioned method. We repeat the process of picking and solving a 2-1 branching structure in one slice until the *m-n branching structure* becomes a *1-n branching structure*. The same process is then applied to the other slice until the structure becomes a *1-1 structure* (simple structure).

Some polygons are formed from several polygons via additional directed edge pairs. If a mesh is extracted by a naive greedy algorithm that produces edges with minimum length at every step, the output mesh will be in an ill form (see Fig. 2(b)). To make the mesh obturated, the mesh has to be extract by the vertex sequence. On the other hand, to guarantee the maximal quality, all possible edges linking the vertices on the two cross-section contours should be considered. Thus, our optimization problem of identifying an energy minizing connectedness pairs perfectly fits into the context of dynamic programming techniques.

There are three kinds of mesh elements in the mesh belt between two polygons on adjacent contours, namely right triangle, invert triangle and quadrangle.

Definition 3. *Begin with two vertices having the shortest distance between two polygons, denotes:*

$pc_{i,j}$ *: the error of a* $right\ triangle_{<i,j>}$ *formed by the ith vertex on the polygon above and the edge* (j,j+1) *on the polygon below, defined as*

$$pc_{i,j} = |p_i, c_j|^2 ; \tag{1}$$

$cp_{i,j}$ *: the error of an* $invert\ triangle_{<i,j>}$ *formed by the edge* (i,i+1) *on the polygon above and the jth vertex on the polygon below, defined as*

$$cp_{i,j} = |c_i, p_j|^2 ; \tag{2}$$

$cc_{i,j}$ *: the error of a* $quadrangle_{<i,j>}$ *formed by the edge* (i,i+1) *on the polygon above and the edge* (j,j+1) *on the polygon below, defined as*

$$cc_{i,j} = 2 * |c_i, c_j|^2 ; \tag{3}$$

$\mathcal{E}_{i,j}$ *: the minimum error of the mesh belt from begin to the ith vertex of the polygon above and the jth vertex of the polygon below, defined as*

$$\mathcal{E}_{i,j} = \begin{cases} 0, & i=0, j=0; \\ \mathcal{E}_{0,j-1} + pc_{0,j-1}, & i=0, j\neq 0; \\ \mathcal{E}_{i-1,0} + cp_{i-1,0}, & i\neq 0, j=0; \\ min\{\mathcal{E}_{i,j-1} + pc_{i,j-1}, & \\ \quad \mathcal{E}_{i-1,j} + cp_{i-1,j}, & otherwise \\ \mathcal{E}_{i-1,j-1} + cc_{i-1,j-1}\} & \end{cases} \tag{4}$$

Note that for the case $i \neq 0$ and $j \neq 0$ in (4), if we set

$$\mathcal{E}_{i,j} = min\left\{\mathcal{E}_{i,j-1} + pc_{i,j-1}, \mathcal{E}_{i-1,j} + cp_{i-1,j}\right\}, \tag{5}$$

the outcome will be triangle-only scheme mesh with maximal quality as showed in Fig. 4(f).

To extract a mesh belt, we first computer all the $\mathcal{E}_{i,j}$, and record the corresponding values from $pc_{i,j}$, $cp_{i,j}$ and $cc_{i,j}$. By backtracking from the end to the beginning, the mesh belt with minimal error (maximal quality) could be extracted (see Fig. 2(c)). Next we scan the whole mesh belt and find out all pairs of mesh elements which involve the additional directed edge pair. If any of these mesh elements is a quadrangle, it will be divided along its shorter diagonal and converted into a triangle. The pair of mesh elements can thus always be converted to the form of two triangles with the additional directed edge being the common edge (see Fig. 2(c)). Finally, the directed edge is replaced by an edge joining the two opposite vertices of the two triangles (see Fig. 2(d)).

After the above process, there may still be some polygons that do not belong to any connectedness pair. Actually, they lie on the top/bottom layers of the real world model or the ends of branches. We simply close these polygons to make the final mesh water tight. To do this, concave polygons are first divided into convex ones. By dividing by its shortest diagonal, each convex polygon will become two smaller convex polygons recursively until all are triangles.

The complete process of extracting mesh from 3D contour structure is summarized in algorithm 1.

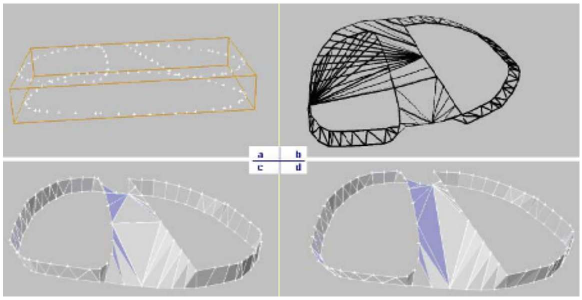

Fig. 2. (a) Contour data with branching structure. (b) Mesh extracted in local optimal by naive greedy algorithm. (c) Mesh extracted in global optimal using the proposed dynamic programming based algorithm. (d) Mesh after removing additional directed edge pair.

Algorithm 1. Mesh Extraction from 3D Rims

```
construct topologically correct cross-section contours;
construct the connectedness pair;
for all connectedness pairs do
    if the structure contains branching structure then
        convert the branching structure into simple structure;
    end if
    compute the minimal error with (4);
    record the choice of computation in each step;
    while backtracking the steps do
        recover the mesh element;
        move backwards;
    end while
    if the structure contains branching structure then
        reconstruct all the elements involving the additional directed edge pair;
    end if
end for
for all polygons not belong to any connectedness pair do
    repeat
        divide the polygon with the shortest diagonal;
    until all are triangles
end for
```

4 Experiments and Results

The first experimental sequence consists of rims recovered from 20 images from a turntable sequence of “Girl and Teddy” toy with fairly complex topology (see Fig. 3). The cameras are calibrated using a method proposed in [12]. The recovered 3D rims are sliced by 121 planes and this results in 6,207 vertices. After reconstruction, the maximal

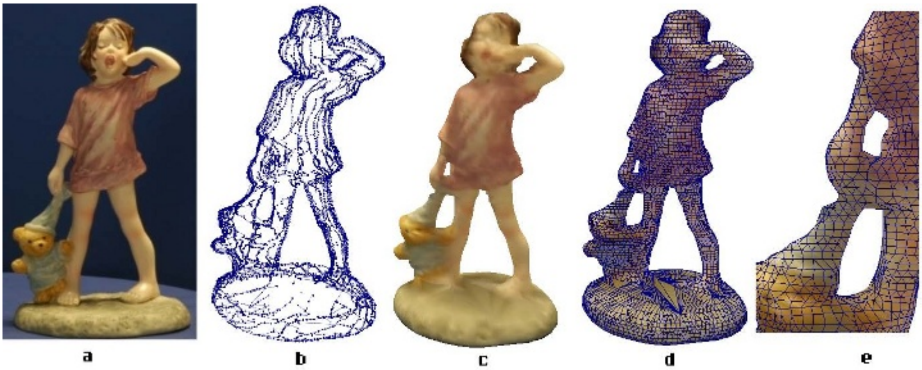

Fig. 3. Reconstruction of a girl and teddy toy with complex topology from a turntable sequence (20 images). (a) The original image. (b) Recovered 3D rims. (c) Resulting surface with texture mapping. (d) Resulting surface with the wire-framed mesh superimposed. (e) Local view of the mesh.

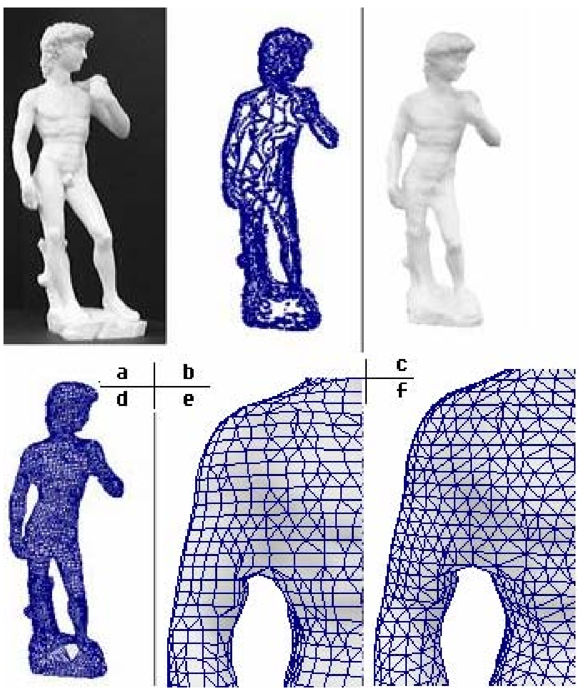

Fig. 4. Reconstruction of a David statuette from a turntable sequence (20 images). (a) The original image. (b) Recovered 3D rims. (c) Resulting surface with texture mapping. (d) Resulting surface with the wire-frame mesh superimposed. (e) Local view of mesh under triangle/quad mixed scheme. (f) Local view of mesh under triangle-only scheme.

quality mesh is formed by 5,805 triangles and 3,334 quadrangles. Figure 3(c) shows the final result after texture mapping from the original image.

The second experimental sequence consists of 3D rims recovered from 18 images of a turntable sequence of a David statuette (see Fig. 4). The recovered 3D rims are sliced by 110 contour planes and this results in 5,990 vertices. Figure 4(c) shows the final result after texture mapping from the original image. Figure 4(e) shows the optimal mesh under mixed scheme with 5,990 triangles and 3,116 quadrangles, comparing with the optimal mesh under triangle-only scheme with 11,980 triangles (see Fig. 4(f)).

5 Conclusions and Future Work

In this paper, we present a novel method for extracting a surface from 3D rims recovered from silhouettes. This method exploits the connectivity information implied by the rim curves to produce a set of topologically correct cross-sections, from which the final surface is extracted. The final surface is a mixed triangle/quadrangle scheme optimal mesh, which produces more regular yet feature-preserving meshes than using the traditional triangle-only mesh.

One limitation is that the geometric position of each vertex is fixed a priori to mesh extraction. Thus, the four vertices of a quadrangle obtained in the reconstruction may not be co-planar. We are now exploring a new algorithm to handle this problem which subdivides the extracted mesh according to the curvature and other local properties of the surface.

Acknowledgements

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

References

1. Liang, C., Wong, K.-Y. K.: Complex 3d shape recovery using a dual-space approach. In: Proc. IEEE International Conference on Computer Vision and Pattern Recognition. Volume 2. (2005) 878–884
2. Boyer, E., Berger, M.: 3d surface reconstruction using occluding contours. International Journal on Computer Vision **22** (1997) 219–233
3. Vaillant, R., Faugeras, O.: Using extremal boundaries for 3-d object modeling. IEEE Trans. Pattern Analysis and Machine Intelligence **14** (1992) 157–173
4. Cong, G., Parvin, B.: Surface recovery from planar sectional contours. In: Proc. International Conference on Pattern Recognition. Volume IV. (2000) 106–109
5. Boissonnat, J.: Shape reconstruction from planar cross sections. Computer Vision Graphics and Image Processing **44** (1988) 1–29
6. O'Neill, B.: Elementary Differential Geometry. 2rd edn. Academic Press (1966)
7. Devillers, O.: On deletion in delaunay triangulations. In: Proc. 15th Annual Symposium on Computational Geometry. (1999) 181–188
8. Schaefer, S., Warren, J.: On c2 triangle/quad subdivision. ACM Trans. Graph. **24** (2005) 28–36

9. Stam, J., Loop, C.: Quad/triangle subdivision. Computer Graphics Forum **22** (2003) 79–85
10. Bresler, Y., Fessler, J., Macovski, A.: A bayesian approach to reconstruction from incomplete projections of a multiple object 3d domain. IEEE Trans. Pattern Analysis and Machine Intelligence **11** (1989) 840–858
11. Meyers, D., Skinner, S., Sloan, K.: Surfaces from contours. ACM Trans. Graph. **11** (1992) 228–258
12. Wong, K.-Y. K., Cipolla, R.: Structure and motion from silhouettes. In: Proc. 8th IEEE International Conference on Computer Vision. Volume II. (2001) 217–222