that is based on two novel representations: the rim mesh, which describes the connectivity of contour generators on the object surface; and the visual hull mesh, which describes the exact structure of the surface of the solid formed by intersecting a finite number of visual cones. We describe the topological features of these meshes and show how they can be identified in the image using epipolar constraints. These constraints are used to derive an image-based practical reconstruction algorithm that works with weakly calibrated cameras. Experiments on synthetic and real data validate the proposed approach.

1. Introduction

Most algorithms for surface reconstruction from outlines compute some form of the *visual hull* [10], or the intersection of solid visual cones formed by back-projecting silhouettes found in the input images. The basic approach dates back to Baumgart's 1974 PhD thesis [1], where a polyhedral visual hull is constructed by intersecting the viewing cones associated with polygonal silhouettes. Volume intersection has remained the dominant paradigm for decades, implemented using representations as diverse as octrees [15] and triangular splines [14]. More recently, graphics researchers have presented efficient algorithms that avoid general 3D intersections by taking advantage of epipolar geometry [11, 13]. Given an image sequence from a camera undergoing a continuous motion, it is also possible to avoid explicit intersections by reconstructing the visual hull as the envelope of the surface tangent planes along the smoothly deforming occluding contours [2, 3, 16].

Defined in full generality, the visual hull is the maximal shape consistent with an object's silhouettes as seen from any viewpoint in a given region, and the *exact* visual hull is the visual hull with respect to a continuous region of space surrounding the object [10]. In this paper we do not treat such limiting cases, but consider the visual hull associated with a finite number of isolated viewpoints (in

on its surface are visual cone patches, edges are intersec
curves between two viewing cones, and vertices are is
points where more than two faces meet. In this cont
visual hull is exact when it correctly captures the co
tivity of these features. Based on this notion of exact
hulls, we specify a novel reconstruction algorithm tha
not rely on polyhedral intersections or voxel-based ca
and produces precise topological and geometric mesh

2. Preliminaries

We assume that we are observing a solid object with w
calibrated pinhole cameras. The surface of the obj
smooth and without planar patches, and the camera
in general position. It is also assumed that apparen
tours of the object have been identified in each input
and oriented counterclockwise, so that the image of th
ject always lies to the left of the contour. To simpli
presentation, we also assume that contours do not co
singularities such as T-junctions and cusps, and restri
attention to objects of genus 0.

In the rest of the paper, we use the following term
ogy. The *rim* or *contour generator* associated with a ca
is the set of all surface points where the optical ray th
the pinhole grazes the object. For general viewpoint
rim is a smooth space curve without singularities [4].
rims can intersect at isolated points on the surface,
frontier points [3, 8, 12], where the viewing rays from
pinholes lie in the surface tangent plane. The proje
of a rim onto the image plane of a camera is the *app*
contour. The set of rays from one camera center pa
through points on the surface forms the *visual cone* a
ated with that camera. As described in the introductio
solid formed by the intersection of all given viewing
is the *visual hull*. Note that the shape of the viewing
depends only on the camera center and on the shape
object, not on the position of the image plane. Thus
jective geometry is sufficient to describe the structure

converge to create a characteristic X-shape (see Figure 1).

Figure 2 shows an example of an ovoid observed by three cameras. Three viewing cones are drawn, along with intersection curves and rims. The figure shows frontier points and *triple points* where three viewing cones intersect. In the figure, each frontier point is incident to four rim arcs and four intersection curve branches, and each triple point is incident to six intersection curve branches, only three of which belong to the visual hull. It can be shown that these incidence relations hold in general.

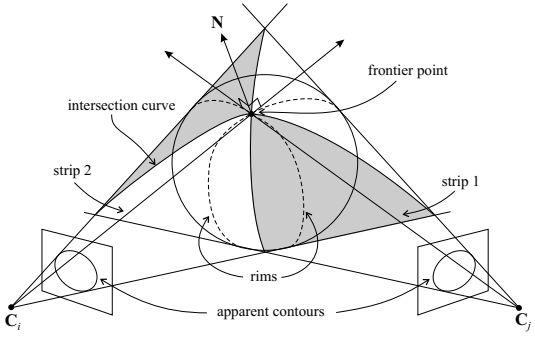

Figure 1: A surface observed by two cameras. The two rims intersect at the frontier point where two cone strips cross.

Now we introduce the two meshes computed by our algorithm. The *rim mesh* is defined on the surface of the actual object. Its vertices are frontier points, edges are rim segments between two successive frontier points, and faces are regions of the surface bounded by two or more edges. A conceptual precursor of the rim mesh is the *epipolar net* of Cross and Zisserman [5], who informally discuss, but do not construct, the arrangement of rims on the surface of an object as the camera moves. The *visual hull mesh* is a topological description of the configuration of visual cone patches on the surface of the solid formed by the intersection of all given visual cones. Its vertices are frontier points (where two strips cross) and triple points (where three cones intersect), edges are intersection curve segments between

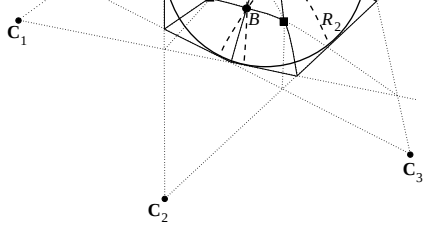

Figure 2: Configuration of rims and intersection curves ovoid observed by three cameras. Rims are dashed arcs, f points are labeled dots, and triple points are squares. Dotte are intersection curve branches outside the visual hull.

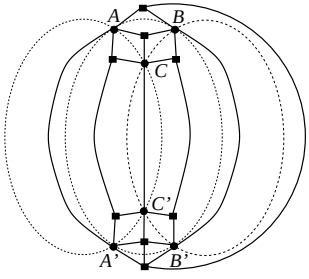

Figure 3: The rim and visual hull meshes of the ovoid from 2. Frontier points are circles, and triple points are squares segments (edges of the rim mesh) are dashed. Intersection segments (edges of the visual hull mesh), are bold lines. No frontier points A', B', and C' belong to the region of the surfa visible in the previous figure.

3. Computing the Rim Mesh

We begin by computing the frontier points, which a vertices of the rim mesh. At a frontier point $\mathbf{P}_{ij}$ c views i and j, the tangent plane to the surface is als epipolar plane determined by $\mathbf{P}_{ij}$ and the camera c $\mathbf{C}_i$ and $\mathbf{C}_j$. In the images, this means that correspo epipolar lines $\mathbf{l}_{ij}$ and $\mathbf{l}_{ji}$ are both tangent to the resp contours at the projections $\mathbf{p}_i$ and $\mathbf{p}_j$ of $\mathbf{P}_{ij}$. Figure lustrates this basic setup, along with other notation th will need later. Finding a pair of matching frontier poi images i and j is a one-dimensional search problem, v

that the corresponding line $\mathbf{l}_{ji}$ in the jth image is tangent to the jth contour. In the presence of contour extraction and calibration errors, there may not exist a pair of matching epipolar lines that exactly satisfy the tangency constraint. In this situation, we find approximately matching frontier points such that the angle difference between the tangent line in one image and the reprojected epipolar tangent from the other image is minimized. Difficulties caused by data error will be further discussed in Section 4.

We obtain all frontier points by matching their projections in two images. The four rim edges incident to a particular frontier point are given by intervals on the two apparent contours that are incident to this point. Thus, there exists a one-to-one correspondence between contour segments in the images and edges of the rim mesh. The orientation of rim edges is then given by the orientation of the corresponding contour segments. In this way, we obtain the complete adjacency information for edges and vertices of the rim mesh. To compute the faces, we need to know the relative ordering in space of the four rim segments incident on each vertex. More formally, we associate with each frontier point $\mathbf{P}_{ij}$ a circular list I where the four edges appear in CCW order around the surface normal. Let $\mathbf{T}_i$, $\mathbf{T}_j$ be the tangents to the rims R_i and R_j at $\mathbf{P}_{ij}$ (see Figure 4). Then the index i appears before the index j in the ordered list I iff

$$(\mathbf{T}_i \wedge \mathbf{T}_j) \cdot \mathbf{N} > 0, \tag{1}$$

where $\mathbf{N}$ is the outward-pointing surface normal (computed as the cross product of the oriented tangent to the contour and the viewing direction).

Equation (1) gives rim ordering in terms of of tangents to the apparent rims, which cannot be computed given only image information. Therefore, we need an equivalent image-based expression for rim ordering. Consider image i and let $\mathbf{v}$ be the direction from the epipole $\mathbf{e}_{ij}$ to $\mathbf{p}_i$, the projection of $\mathbf{P}_{ij}$. Let also $\mathbf{t}$ be the tangent to the contour at $\mathbf{p}_i$ (see Figure 4), and let k_s be the apparent curvature at the surface is elliptic at $\mathbf{P}_{ij}$; or (b) the camera rotate and the surface is hyperbolic. The sign of $\mathbf{v} \cdot \mathbf{t}$ is posi the camera rotates CCW in the tangent plane and ne otherwise. Moreover, the sign of the apparent curvatu is positive if the surface is elliptic or negative if the su is hyperbolic [9]. Thus, expression (2) is positive in th above mentioned cases and negative otherwise. It is fore the desired image-based expression equivalent to

Given the above rim ordering criterion, it is straig ward to trace the loops of edges bounding rim faces. pose we start with one rim segment s of the rim R_i and to find the face that lies to its left. Informally, we tra s along its direction and at its endpoint $\mathbf{P}_{ij}$, simply left turn to get to the next edge, that is, we select the preceding s in the ordered circular list of $\mathbf{P}_{ij}$. We from endpoint to endpoint in this manner, traversing either forward or backward along their orientation, tal left turn each time until we complete a cycle.

4. Computing the Visual Hull Mes

As demonstrated in the previous section, we can pute the topology of the rim mesh without knowing thing about its geometry, except for the positions of tier points (note that under weak calibration we can recover these positions up to a projective transforma This topology constrains the adjacency relationship tween triple points and intersection curves, which a vertices and edges of the visual hull mesh.

A triple point $\mathbf{P}_{ijk}$ the intersection of three optica back-projected from contour points $\mathbf{p}_i$, $\mathbf{p}_j$, and $\mathbf{p}_k$ in different images (see Figure 5). In particular, $\mathbf{p}_k$ sa the *transfer equation* [7]

$$\mathbf{p}_k = \mathbf{l}_{ki} \wedge \mathbf{l}_{kj} = (\mathbf{F}_{ik}\,\mathbf{p}_i) \wedge (\mathbf{F}_{jk}\,\mathbf{p}_j),$$

where $\mathbf{F}_{mn}$ is the *fundamental matrix* mapping poi image m to epipolar lines in image n, and homoge

Figure 5: The triple point $\mathbf{P}_{ijk}$ is a "phantom point" where the rays formed by back-projecting epipolar correspondents $\mathbf{p}_i$, $\mathbf{p}_j$, and $\mathbf{p}_k$ meet in space. $\mathbf{P}_{ijk}$ can be located by tracing the intersection curve Γ_{ij} between views i and j and noticing when its projection γ_{ij} in the kth image crosses the contour.

coordinates are used for image points. Points $\mathbf{p}_i$ and $\mathbf{p}_j$ satisfy symmetric equations. Any pair of $\mathbf{p}_i$, $\mathbf{p}_j$ and $\mathbf{p}_k$ are *epipolar correspondents* — that is, any two of the points lie in the epipolar plane defined by the two camera centers and one of the points [2]. A triple point is a standard trinocular stereo correspondence, but it does not usually lie on the surface because $\mathbf{p}_i$, $\mathbf{p}_j$ and $\mathbf{p}_k$ are projections of three different points $\mathbf{P}_i$, $\mathbf{P}_j$, and $\mathbf{P}_k$ on three different rims.

Just as with finding frontier points, finding triple points is a one-parameter search. We walk along the ith contour in discrete steps and for each contour point $\mathbf{p}_i$ find the epipolar line $\mathbf{l}_{ji} = \mathbf{F}_{ij}\,\mathbf{p}_i$ in image j, and locate an epipolar correspondent $\mathbf{p}_j$ by intersecting $\mathbf{l}_{ji}$ with the jth contour. We then obtain a third point $\mathbf{p}_k$ by transferring $\mathbf{p}_i$ and $\mathbf{p}_j$ using (3) and check whether $\mathbf{p}_k$ lies on the kth contour. As shown in Figure 5, transfer of successive epipolar correspondents allows us to trace the intersection curve Γ_{ij} between ith and jth cones in the kth image. The triple point is revealed when the traced curve crosses the kth contour.

In general, epipolar correspondents are not unique. In the case shown in Figure 5, each epipolar line intersects each contour twice (this reflects the fact that intersection curves have multiple branches). Moreover, the epipolar correspondence criterion does not say when a triple point belongs to the visual hull — the additional constraint is that the point must not project outside the silhouette in any other input view. However, if we are able to exactly compute the positions of frontier points along the contours, we can use this information to simplify the search for triple points.

Each face of the rim mesh is bounded by rim segments that also belong to the surface of the visual hull, so we can

intersection curve, we can identify all the segments b
ing a cone strip. Individual faces of the strips are ide
by grouping all the intersection curve segments that p
within the contour interval that corresponds to a part
rim segment.

The algorithm described above yields the correct
hull mesh given exact input data (perfectly extracted
tours, error-free fundamental matrices). However, nois
calibration error tend to destroy exact topological fea
most importantly, intersection curve crossings at fr
points. Figure 6 illustrates the situation. The epipola
$\mathbf{l}_{ij}$ is tangent to the contour at point $\mathbf{p}_i$ in the ith image
line corresponds to the line $\mathbf{l}_{ji}$ in the jth image, which
tangent to the jth contour, but intersects it in two ep
correspondents $\mathbf{p}_{j1}$ and $\mathbf{p}_{j2}$. Intersecting the visua
due to these three points in the epipolar plane yield
distinct intersection curve points $\mathbf{P}_{ij1}$ and $\mathbf{P}_{ij2}$, inste
a single frontier point. Thus, instead of being singula
intersection curve separates into two distinct branche
do not meet. In order to approximate the position of a
tier point, we have to match $\mathbf{p}_i$ with the epipolar tan
point $\mathbf{p}_j$ in the jth image. However, the two points c
lie in the same epipolar plane, and visual rays through
do not intersect. We could estimate the location of $\mathbf{I}$
the midpoint of the segment connecting the points of c
approach of the two rays, but this approximated poin
not lie on the traced intersection curves. This leads
rious consistency problems for a naive implementatio
attempts to strictly enforce combinatorial constraints
act visual hull structure.

Intuitively, small perturbations to exact contour an
ibration data result in contours that back-project to g
cones in space, the intersection of which does not ha
share the properties of exact visual hulls. For ins
while we know that cone strips never break up in t
(each ray interval along the strip must contain at lea
point), in noisy data, they may break up near the fr
points as shown in Figure 6. Nevertheless, even with

rate faces of the rim mesh, and then clip out all components of the curves that project outside any of the silhouettes. In the process, combinatorial information about the curves is maintained, so that it becomes possible to recover the geometry of cone strips. Namely, boundary points of the strips are connected in the order induced by the contour parametrization, and are separated into two groups that correspond to segments bounding the rim on the near and the far side with respect to their distance from the camera. This data structure is a monotone polygon, and it can be triangulated in linear time for purposes of display.

5. Experimental Results

The first input sequence consists of six synthetically generated images of an egg model. Contours were extracted using snakes and modeled as cubic B-splines. As seen in Figure 7, the algorithm correctly generates the rim mesh and the visual hull, both in their topological and geometric form. The contrast between these two forms is clearly visible by comparing two renderings of the same strip in Figure 7 (f) and (g). The exact strip explicitly shows frontier points and triple points, and intersection curves are forced to converge in four branches at frontier points. The triangulated strip does not degenerate at frontier points, because the robust strip tracing algorithm ignores them.

We also demonstrate results for two calibrated nine-image turntable sequences of a gourd and a vase (Figure 8). For both of these data sets, the algorithm constructs a complete rim mesh, even though many of the frontier points are densely clustered near the top and the bottom. To better visualize the structure of these meshes, we rendered their vertices and edges as graphs using a publicly available graph drawing program. The graphs, shown in Figure 8 (b) and (h), reveal the regular structure of rim crossings which is impossible to observe in the images themselves. Each rim mesh in our examples obeys Euler's formula for topological polyhedra of genus 0, $V + F = E + 2$ (note that the

6. Discussion and Future Work

Our preliminary results are intriguing. Significantly, t
covery of exact rim meshes has proven to be robust
with densely clustered frontier points that do not
matching epipolar lines. Note that the rim mesh
ture depends only on the relative ordering of frontier
along the rims, not on absolute positions — hence the
tive stability of the topology. With visual hull meshe
situation is different: the connectivity of intersection c
and triple points is elusive, while the geometry ma
be recovered reliably. It will be important to inves
the question of whether these instabilities are inher
the conditioning of exact visual hull computation,
introduced by our algorithm. We are considering a
ent approach to recovering the topology of the visua
mesh that would take full advantage of the combina
constraints given by the structure of the rim mesh. W
also working on extending our implementation to dea
T-junctions and surfaces of arbitrary genus, to handle
complex and visually interesting objects.

Overall, our approach has several attractive fea
Most importantly, it is based on an analysis of the
structure of visual hulls from finitely many viewp
which has received little attention in previous research
approach takes advantage of epipolar geometry and
calibration — in a sense, we don't need to know
the cameras are. Moreover, our algorithm uses only
dimensional computations, and constructs a range of
representations, from graph-theoretic and topologic
completely image-based, to purely geometric. For
reasons, it brings fresh insights to the theory and pr
of the venerable problem of visual hull computation.

Acknowledgments. This work was supported in part
Beckman Institute, the National Science Foundation
grant IRI-990709 and a UIUC-CNRS collaboration a
ment.

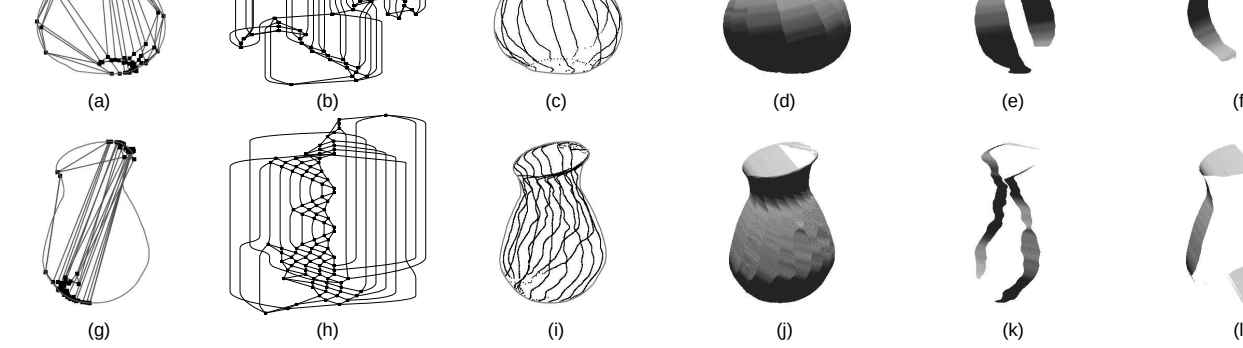

Figure 8: Top row: gourd results. (a), (b) the rim mesh and corresponding graph (96 vertices, 192 edges, 98 faces); (c) intersection that make up the visual hull; (d) triangulated visual hull model; (e), (f) two of the strips from the model. Bottom row: teapot results. (g), mesh (104 vertices, 208 edges, 106 faces); (i) intersection curves on the surface of the visual hull; (j) visual hull model; (k), (l) two cone

References

[1] B.G. Baumgart, "Geometric Modeling for Computer Vision", Ph. D. Thesis (Tech. Report AIM-249), Stanford University, 1974.

[2] E. Boyer and M. Berger, "3D Surface Reconstruction Using Occluding Contours", *Int. J. Comp. Vision*, 22(3), 1997, pp. 219-233.

[3] R. Cipolla, K. Astrom, and P.J. Giblin, "Motion from the Frontier of Curved Surfaces", *Proc. IEEE Int. Conf. on Comp. Vision*, 1995, pp. 269-275.

[4] R. Cipolla and P.J. Giblin, *Visual Motion of Curves and Surfaces*, Cambridge University Press, Cambridge, 1999.

[5] G. Cross and A. Zisserman, "Surface Reconstruction from Multiple Views Using Apparent Contours and Surface Texture", *NATO Advanced Research Workshop on Confluence of Computer Vision and Computer Graphics*, 2000, pp. 25-47.

[6] M. do Carmo, *Differential Geometry of Curves and Surfaces*, Prentice-Hall, Englewood Cliffs, New Jersey, 1976.

[7] O. Faugeras and L. Robert, "What Can Two Images Tell Us About a Third One?", *Int. J. Comp. Vision*, 18(1), 1996, pp. 5-19.

[8] P.J. Giblin and R. Weiss, "Epipolar Curves on Surfaces", *Image and Vision Computing*, 13(1): pp. 33-44, 1995.

[9] J. Koenderink, "What Does the Occluding Contour Tell Us Solid Shape?", *Perception*, 1984, pp. 321-330.

[10] A. Laurentini, "The Visual Hull Concept for Silhouette-bas age Understanding", *IEEE Trans. on Pattern Analysis and M Intelligence*, 16(2), 1994, pp. 150-162.

[11] W. Matusik, C. Buehler, R. Raskar, S. Gortler, and L. Mc "Image-based Visual Hulls", *Proc. SIGGRAPH* 2000, pp. 369

[12] J. Porrill and S. Pollard, "Curve Matching and Stereo Calib *Image and Vision Computing*, 9(1): pp. 45-50, 1991.

[13] I. Shlyakhter, M. Rozenoer, J. Dorsey, and S. Teller, "Reco ing 3D Tree Models from Intrumented Photographs", *IEEE Co Graphics and Applications*, 21(3), 2001, pp. 53-61.

[14] S. Sullivan and J. Ponce, "Automatic Model Construction, P timation, and Object Recognition from Photographs using Tri Splines", *IEEE Trans. on Pattern Analysis and Machine Intell* 20(10), 1998, pp. 1091-1096.

[15] R. Szeliski, "Rapid Octree Construction From Image Sequ *CVGIP: Image Understanding*, 1(58), 1993, pp. 23-32.

[16] R. Vaillant and O. Faugeras, "Using Extremal Boundaries f Object Modeling", *IEEE Trans. on Pattern Analysis and Mach telligence*, 14(2), 1992, pp. 157-173.