Chapter 1

SURFACE RECONSTRUCTION FROM MULTIPLE VIEWS USING APPARENT CONTOURS AND SURFACE TEXTURE

Geoffrey Cross and Andrew Zisserman

Abstract We describe a novel approach to reconstructing the complete surface of an object from multiple views, where the camera circumnavigates the object. The approach combines the information available from the apparent contour with the information available from the imaged surface texture.

It is demonstrated that this approach of combining two information sources has significant advantages over using either the contour or texture alone: first, the geometric constraints available are complementary, so that the deficiencies of one source can be overcome by the strengths of the other; second, judicious use of the two sources enables an efficient automatic reconstruction algorithm to be developed.

In particular we make the following contributions: it is shown that the set of epipolar tangencies generates an *epipolar net* at which the visual hull coincides with the surface; a statistical cost function is defined which incorporates terms for error in image intensity similarity and geometric error in the apparent contour; and, an improved photo-consistency constraint is developed for space carving.

Examples of automatically generated texture-mapped graphical models are given for various objects and camera motions. The objects may contain concavities, and have non-trivial topology.

1. INTRODUCTION

Surface reconstruction from images has been extensively investigated over the past three decades. The methods that have been developed can broadly be categorized into four classes: reconstruction from the apparent contour [5, 7, 13, 15, 18, 30, 32, 33] where the *visual hull* [21] is computed; texture correlation [2, 10, 14, 16] where a dense reconstruction is computed based on a measure of intensity similarity; feature based matching, e.g. [1, 24] which produces only a sparse surface map; and, more recently, space carving [20, 27] which is a variant

on texture correlation, where voxels are progressively carved from an occupancy representation based on the principle of "photo-consistency" between images.

A secondary issue, but of considerable importance in implementations, is how the surface is represented. The choice is broadly either a parametrized surface, for example a triangulated mesh or tensor product surface; or voxel occupancy. A recent elegant addition is representation by a level set of a hypersurface in 4D [28]. This representation was the basis of the texture correlation method described in [10], and has the advantage of being able to represent surfaces with various topologies.

The objective of this paper is the automatic reconstruction of a surface from multiple views, where the camera may circumnavigate an object. This means that methods which build a depth-map reconstruction from a particular view are not appropriate. The novelty here lies in combining elements from the various classes of reconstruction method, and also in an implementation which uses different surface representations as appropriate for accuracy and efficiency, such that all views contribute equally.

We start in section 2.1 by rehearsing and extending the geometry of the visual hull, which is the surface bound computed by back-projecting apparent contours. In particular we introduce the "epipolar net" which is the collection of surface point constraints provided by the apparent contours over a set of view points. The computation of the visual hull is then described in section 2.3. Section 3. introduces space carving and the photo-consistency test which is central to the algorithm. We then develop and implement an extension to the standard photo-consistency test [19]. The extension enables sub-pixel registration and greater invariance to photometric variations. Section 4. then describes a statistical cost function which is an extension of that proposed and implemented by Faugeras and Keriven in [10]. The extension is to include a geometric error based on conformity to the measured apparent contour.

These three developments (visual hull, photo-consistency, and statistical optimization of the surface) form the elements of an efficient and 'optimal' automatic surface reconstruction algorithm in which space carving proceeds from the visual hull, rather than the much looser bound usually used in [20], and the statistical optimization proceeds from the space carved surface. Because the cost function includes both texture and apparent contour terms, regions of the surface may be reconstructed even in the absence of adequate texture for correlation.

It is assumed that the camera matrices are known for each view. For all the example sequences included here the objects are rotated on an (uncalibrated) turn table, and the camera matrices are computed automatically from the image sequence using the method described in [11]. Other methods for generating the cameras from corner features, e.g. [3, 12, 34], or from the apparent contour and other image features [4, 6, 8, 25] could equally well be used.

Figure 1.1 Six images (*top* image set) from a sequence of a toy dinosaur. The figure demonstrates that the apparent contours (*middle* image set) are a rich source of information for reconstruction. Corner matches (*lower* image set) also provide further information on the surface structure.

2. RECONSTRUCTION FROM APPARENT CONTOURS

In this section we first review the geometric constraints on surface reconstruction provided by the apparent contour over multiple images. Then we describe a surface reconstruction algorithm using the apparent contours.

2.1 GEOMETRY

The image outline of a smooth surface S results from surface points at which the imaging rays are tangent to the surface, as shown in figure 1.2. The *contour generator* on S is the set of points $\mathbf{X}$ on S for which rays are tangent to S. The corresponding image *apparent contour* is the set of points $\mathbf{x}$ which are the image of $\mathbf{X}$, i.e. the apparent contour is the image of the contour generator. Image points on the apparent contour back-project to rays tangent to the surface, and image lines tangent to the apparent contour back-project to planes which are tangent planes to the surface.

Back-projecting apparent contours. For known cameras, the apparent contour generates a set of tangency constraints and a bound for surface reconstruction. Each apparent contour back-projects to a "cone" as shown in figure 1.3(a). The surface is tangent to this cone at the contour generator, and all points on the surface must lie on or within this semi-infinite cone.

If two or more images of the surface are available, the cones intersect to enclose the surface [17]. The closed surface as reconstructed from these cones

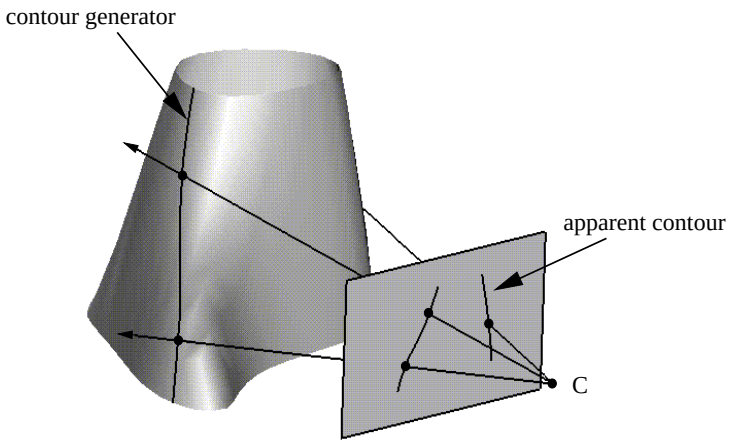


Figure 1.2 The apparent contour is the image of the contour generator. The surface tangent plane at any point on the contour generator passes through the camera centre $\mathbf{C}$.

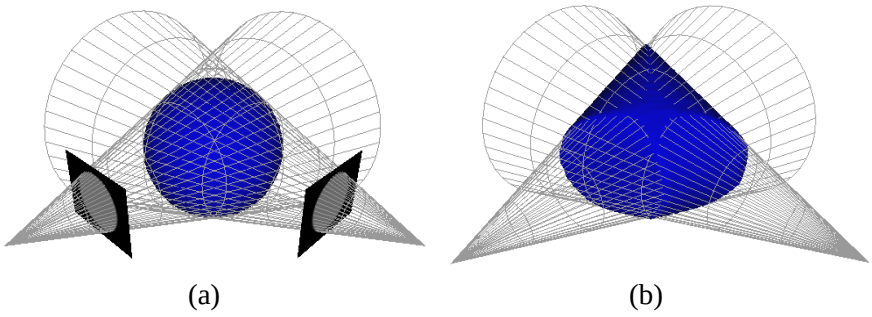


Figure 1.3 (a) Back-projecting the apparent contour gives a cone tangent to the surface along the contour generator. (b) The visual hull is given by the intersection of the cones, one for each view. It encloses the original surface and is tangent to this surface along the contour generators.

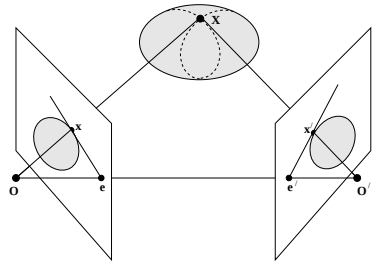


Figure 1.4 At an epipolar tangent point **X** on the surface, the surface tangent plane passes through both camera centres.

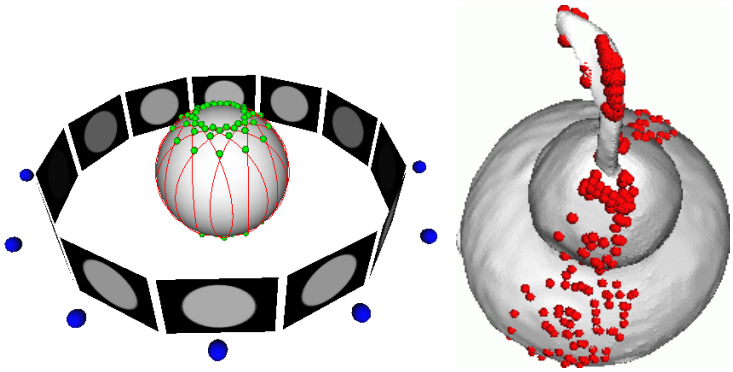
is referred to as the *visual hull* (see figure 1.3(b)). The visual hull is guaranteed to fully enclose the surface (as each of the generating cones enclose the surface). The surface will be coincident with and tangent to the visual hull along each of the contour generators but not elsewhere. As can be seen in figure 1.6(a), the visual hull does not, however, penetrate concavities of the surface, and only reconstructs the convex elliptic and hyperbolic parts of the surface.

Epipolar tangents. Consider two views. In general the cones back projected from apparent contours intersect in a space curve which does not lie on the surface. However, as shown in figure 1.4, at a point where the epipolar plane is tangent to the surface the space curve is coincident (and tangent to) the surface. Such points arise where the apparent contour is tangent to the epipolar lines, and are termed *epipolar tangencies* [25, 26] or *frontier points* [6].

Epipolar nets. Since an epipolar tangent point on the surface arises from contact with an epipolar plane, it is a property of view pairs. If one of the camera centres changes position then the epipolar tangent points will also move in general[1]. For three views there are epipolar tangencies for each pair of images of the triplet, and over a set of views an *epipolar net* of such points is built up (see figure 1.5). Each point of the net arising at the intersection of two contour generators for a view pair. However, since not all contour generators actually intersect with each other, some image pairs will not provide epipolar tangent points.

Each point (node) on the epipolar net is a 3D point which can be found directly from the apparent contour. The visual hull is coincident with the viewed surface at each of the 3D points of the net. Therefore in areas with a dense epipolar net the visual hull models a surface more closely than elsewhere.

[1]Epipolar tangent points are the same over multiple views if the cameras share an epipolar plane, e.g. in the case of three views with collinear camera centres.



(a) (b)

Figure 1.5 The epipolar net. (a) The intersections of each pair of contour generators are epipolar tangent points for the associated pair of images. The epipolar tangent points combine to create an epipolar net of points through which both the surface and the visual hull must pass. The camera centres and image planes are shown. (b) The model from the gourd sequence of figure 1.15 with all epipolar tangent points superimposed demonstrates that a significant part of the surface is covered by the epipolar net. The net is for 30 images covering a 180 degree sweep.

2.2 DEFICIENCIES IN RECONSTRUCTING FROM THE VISUAL HULL

As shown above, reconstruction from apparent contours alone provides the visual hull which is a bound on the surface, but does not fully reconstruct the surface. There are two types of deficiencies: first, concavities are not reconstructed since they do not contribute to the apparent contour, see figure 1.6(a); second, even for a convex surface the visual hull is not coincident with the surface, see figure 1.6(b). This difference between the visual hull and surface depends on the surface shape, and the number and position of the views.

On the other hand, surface points (e.g. texture), viewed over multiple images and reconstructed by triangulation, "probe" the surface at all visible points including concavities, see figure 1.7. So, triangulating on surface features does reconstruct concavities, but provides no constraints at texture-less surface patches; and, the visual hull does not reconstruct surface concavities, but does reconstruct surface regions even in the absence of any surface features. Consequently, triangulating on surface texture features complements the tangency constraints provided by the apparent contour. We return to triangulation on surface texture in section 3..

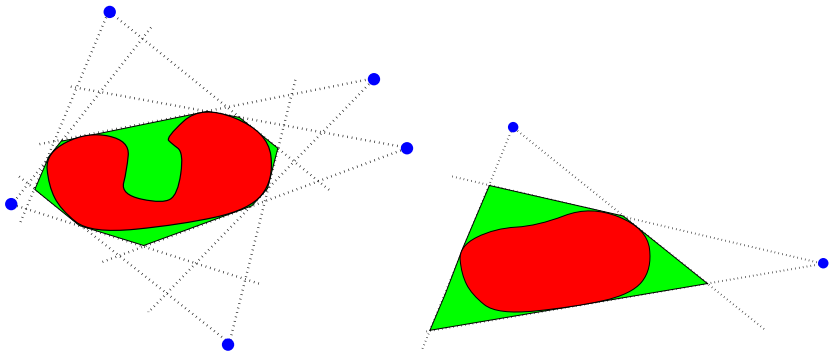


Figure 1.6 2D schematic of the difference between the 'surface' and the visual hull computed from image 'outlines'. (a) The visual hull does not 'capture' surface concavities; (b) Even if there are no concavities, the visual hull computed from a finite number of images may not coincide with the surface.

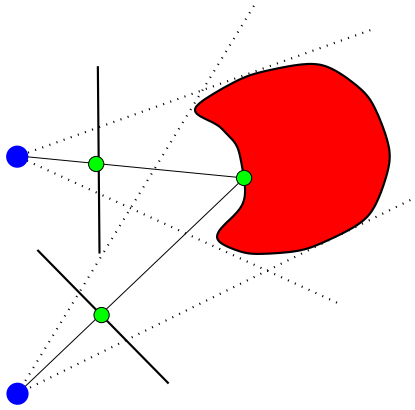


Figure 1.7 Two views of a surface provide information from both the apparent contours in the form of a tangent constraint and a point match which provides a surface point constraint. Both reconstruction methods complement each other, and will be used together to provide complete models.

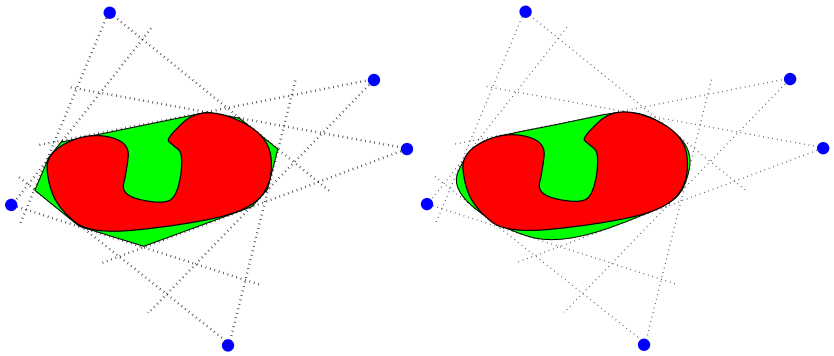


Figure 1.8 (a) The visual hull can be reconstructed as the intersection of the cones generated by back-projecting the apparent contour in each image. (b) If assumptions are made about the smoothness of the surface, the visual hull can be reconstructed by locally fitting quadratic or Bezier patches.

2.3 COMPUTING THE VISUAL HULL

If no assumptions are made about the surface shape, the visual hull must simply be reconstructed as the intersections of the back-projected cones as in figure 1.8(a). However, an approach used by several authors [5, 7, 15] is to assume a local quadric form of the surface which is fitted generally to three apparent contours (see figure 1.8(b)). This hull surface is then parametrized by a net formed by the contour generator for each view linked by epipolar curves. The problem with this parameterization is that it is singular at epipolar tangencies, and reconstruction errors result. An alternative approach is a volumetric computation where the visual hull is computed and represented by voxel occupancy. This idea dates back to [22].

Volumetric model generation. A region of space, known to enclose the surface, is subdivided volumetrically at a given (but not necessarily uniform) resolution. Each sub-region is then classified according to whether it lies inside the visual hull, outside the visual hull or spans the surface of the visual hull (figure 1.9). This classification is achieved by observing the projection of the sub-region, or *voxel*, onto each image plane: if the projected voxel lies outside any of the apparent contours, it must also lie outside the visual hull. The algorithm is close to that presented by Szeliski in [31].

Octree representation. An implementation based on octrees has significant speed advantages. A region of space is subdivided into a set of cubes at a low resolution. Each cube is then classified with respect to the visual hull as above.

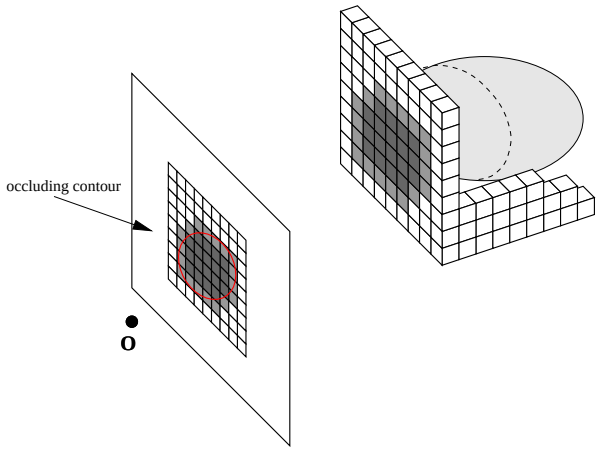


Figure 1.9 A region of space is sampled, and each voxel is assigned one of three values. Light cubes are known to lie outside the visual hull, and dark cubes are known to lie inside the visual hull (from this single image). Gray cubes lie on the surface and can be sub-sampled for greater resolution.

If a higher resolution is required, each cube is subdivided into 8 equal sized cubes and the classification is repeated. However, cubes which are known to lie completely outside or completely inside the visual hull do not need to be subdivided, as the classification of their 'children' will be the same as that of their parents.

Using this approach, an arbitrary volumetric resolution can be achieved. A trade-off is required at this point: at a low resolution, important topological features of the visual hull may be missed, whilst high resolution models take longer to generate and are more difficult to manipulate. Figure 1.10 demonstrates this trade-off.

3. COMPUTING THE SURFACE FROM TEXTURE INFORMATION

It has been shown that the apparent contour does not give any information about concavities of the viewed surface. In order to accurately reconstruct such surface regions, it is necessary to make use of other information such as texture.

Dense stereo algorithms, e.g. [16], make good use of texture information, but to date have only been extended from two views to many by merging surface patches — an approach that does not generalize well to circumnavigation of the object.

It is also possible to obtain surface shape information from features such as corners or curves in the image. However, this data is sparse and generally a

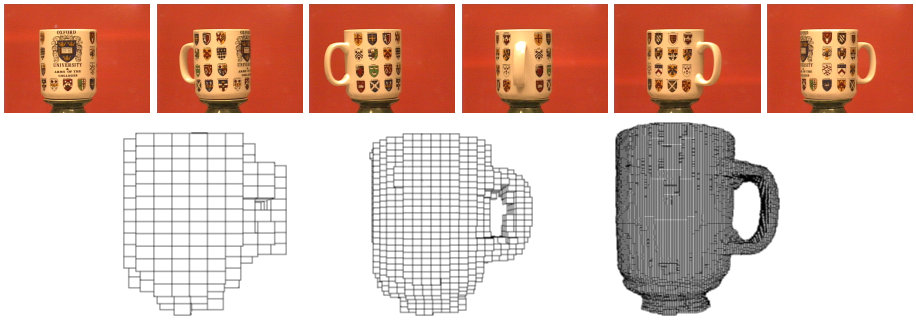


Figure 1.10 Three different resolution reconstructions (from *left* to *right*: 728 voxels, 4k voxels and 246k voxels) of the visual hull of a mug imaged from 36 views (six of the original images are shown in the top row). Note, the change in topology between the first and second resolution.

parametrized surface model must subsequently be fitted, e.g. [14], in order to obtain a dense reconstruction.

3.1 SPACE CARVING

Space carving [20, 27] does not suffer from either of these limitations. It provides a successive algorithm for removing voxels in a voxmap with the aim of creating a 3D shape which reproduces the input images.

In brief, the idea of space carving is to project each voxel on the surface into the set of images. The projected voxel defines a correspondence between image points, and the intensity at the corresponding points is evaluated to determine if it is "photo-consistent" (see below), ensuring the voxel is only compared with pixel values in images in which it is not occluded. If the voxel is not photo-consistent it is removed (carved) from the occupancy space, and the algorithm repeats. If it is photo-consistent then the voxel can be coloured with the corresponding pixel value. The final result is a set of voxels which closely reproduce the original images.

There are 2 main drawbacks to the original implementation described in [19]. The first concerns initialization, and the second the photo-consistency test. The following sections describes improvements in these two areas.

3.2 SURFACE DIRECTED INITIALIZATION

It is often necessary to initialize the algorithm with a very large voxmapped space in order to ensure that it completely encloses the surface. Each voxel must then be tested in turn for consistency with the images which results in a high computational load. As has been shown in section 2.1, the visual hull completely encloses the generating surface, and can be efficiently constructed

using octrees. It follows naturally that the visual hull should be used as a starting point for the space carving algorithm.

3.3 IMPROVED PHOTO-CONSISTENCY CONSTRAINT

In the original implementation of space carving [19], the photo-consistency test proceeded as follows: project a voxel centroid into each image, and compare the intensities of the corresponding pixels. If the intensities differ by less than a threshold than the voxel is photo-consistent. Unfortunately this test is prone to errors from image intensity noise and from the spatial sampling noise inherent in the voxelation of space. A "shift transform" was introduced in [20] to solve the second of these problems. However, it can be shown that a bias is introduced such that the model is larger than the original object and hence misses certain small concavities.

If the threshold on the photo-consistency test is too low, the result is false protrusions on the surface. Conversely, (and more serious here), if the test is too conservative (i.e. points are not considered consistent even when they are) the result is indentations or holes in the final surface. The photo-consistency test must therefore be robust to image noise.

Here we improve the photo-consistency test so that it is more robust in two ways: first, a smooth surface fit is used to provide an image to image map (see figure 1.11). This enables registration to sub-pixel accuracy. Interpolation is now required because pixels from one image will not in general coincide with pixel positions in the other image; second, the comparison of intensities uses normalized cross-correlation so is invariant to a local affine transformation in brightness ($I \rightarrow \alpha I + \beta$). The photo-consistency test defines a binary valued *indicator function* for voxels.

Photo-consistency constraint — implementation. In order to provide an accurate and smooth surface-induced mapping, it is necessary to fit a local surface patch to the voxelated surface. This is achieved by locally fitting a quadratic patch to a set of surface points. Each patch can then be represented as a quadric surface in the form

$$\mathbf{x}^\top Q \mathbf{x} = 0 \ ,$$

where $\mathbf{x} = (X, Y, Z, 1)^\top$ is a 3D point on the surface patch, and Q is a 4×4 symmetric matrix representing the quadric. It can be shown [9, 29] that the quadric induces an algebraic mapping, $\mathbf{x}_i = \mathbf{f}(\mathbf{x}_j)$ between any two images, i and j, where $\mathbf{x}_i$ and $\mathbf{x}_j$ are corresponding points in images i and j respectively.

The photo-consistent indicator function is then computed as follows for each voxel: fit a quadric patch to the voxel neighbourhood, and project the voxel centroid into each image. The projected centroid defines corresponding points

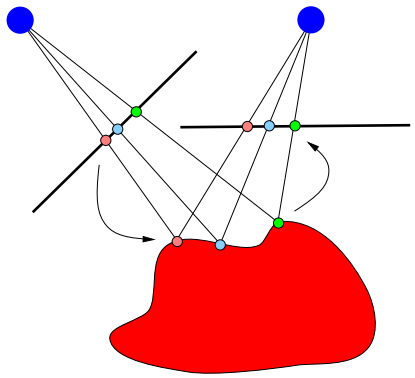


Figure 1.11 Surface-induced transfer. Any image point back-projects to a ray in space. Given the local geometry of a surface, this ray can be intersected with the surface to give a 3D point, which in turn can be projected into a second image. This process defines a point to point mapping between the images induced by the surface.

between the images. Starting from the most fronto-parallel image measure the normalized cross-correlation with corresponding image neighbourhoods, where the map between images is defined by the fitted quadric patch. If the cross-correlations are above a threshold (here 0.8) then the voxel is photo-consistent.

3.4 SPACE CARVING IMPLEMENTATION

The starting point for the space carving algorithm is the voxel representation of the visual hull produced from the apparent contours. Often, and in particular when many views of a single object are available, the visual hull is close to the final surface. Incorrectly modeled regions (the two deficiencies illustrated in figure 1.6) are identified using the indicator function, and only these regions are a candidate for space carving. Voxels are successively removed in these regions until the surface is photo-consistent with the images.

3.5 RESULTS

Figure 1.12 shows six images from a sequence of 36 images of a skull, and the computed visual hull. The eye and nose areas are not accurately modeled by the visual hull. However, areas around the back of the head which are convex and smooth are accurately reconstructed in the visual hull. These two types of regions are identified by the photo-consistency indicator function. Figure 1.13 shows the number of voxels remaining in the model through iterations of the algorithm in section 3.4.

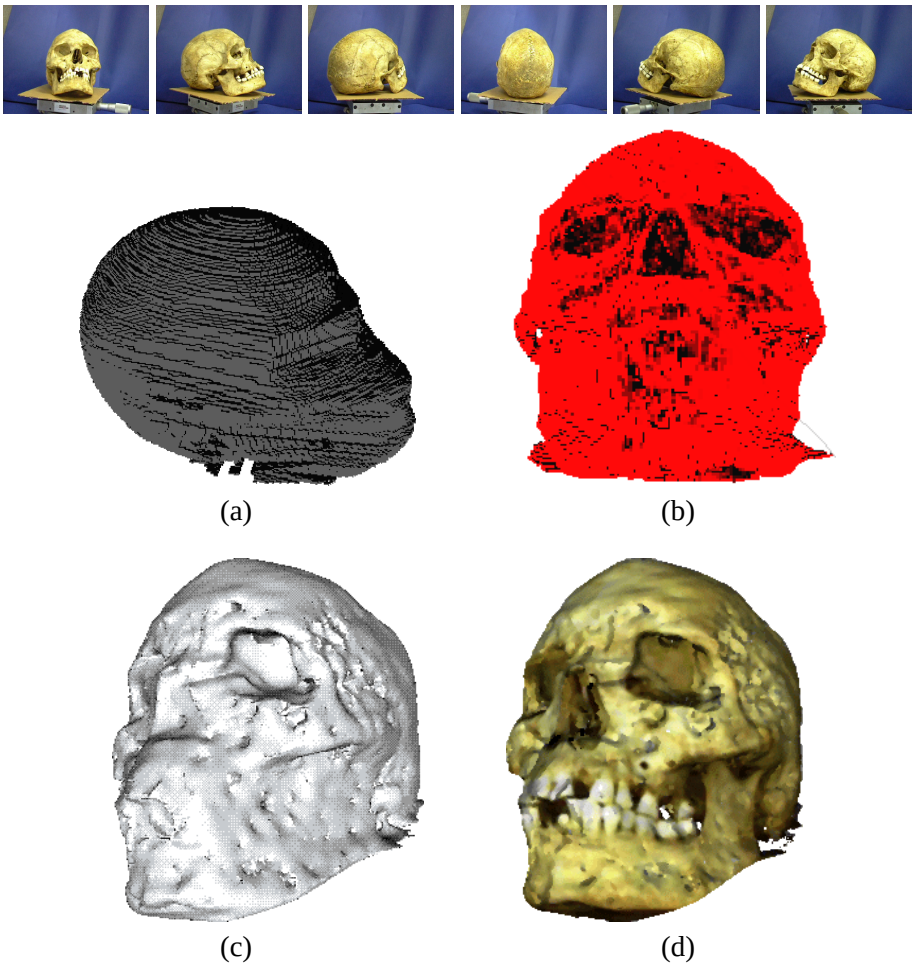


Figure 1.12 (a) A voxelelated model of the visual hull of a skull. It can be seen that the visual hull "smoothes over" concavities such as the nose and eye sockets. (b) An indicator function (section 3.3) is applied to the surface of the visual hull, and regions with high score are shown in *gray* and those with low score are shown in *black*. A high score indicates that the reprojection of the surface is consistent with the input images. It can be seen that the eyes and nose regions have been marked as "incorrect". (c) The final model after applying the space carving algorithm as described in section 3.4. After space carving, areas such as the eye socket and nose region are correctly reconstructed. A simple mesh smoothing algorithm has been applied to the surface to highlight these areas, but the model is stored as 128^3 voxels. (d) A textured-mapped model using intensities from the original images.

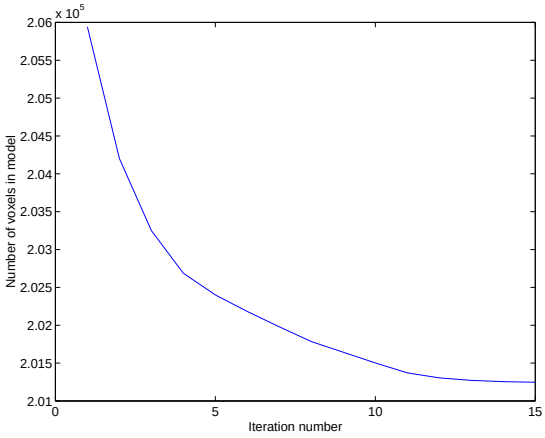


Figure 1.13 Number of voxels remaining in model (figure 1.12) as the space-carving algorithm converges.

Figure 1.14 Texture-mapped 3D models of the dinosaur from sequence in figure 1.1 and the cup from the sequence in figure 1.10

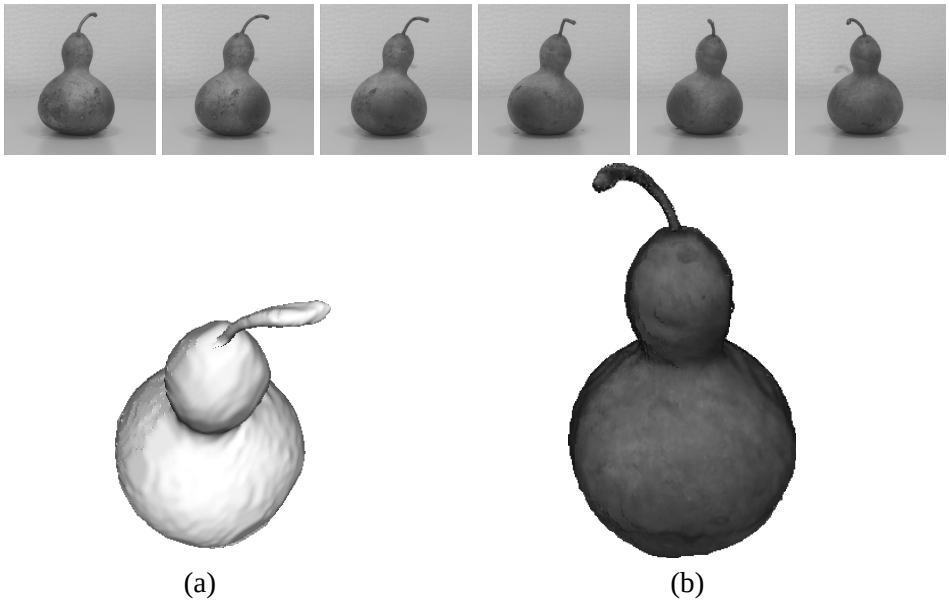


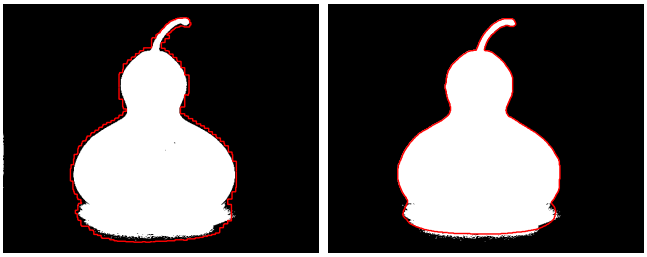
Figure 1.15 (*top*) Six images from a sequence (30 images) of a gourd. (a) An untextured model and (b) a textured model.

Figures 1.14 and 1.15 shows further examples of using the voxel based algorithm. In each case the VRML models generated have in the order of 50k faces which allows them to be rendered in reasonable speeds on a workstation.

4. MINIMISING REPROJECTION ERRORS USING A SURFACE REPRESENTATION

Although the results from the space carving algorithm are subjectively good, and objectively accurate to within one voxel, they are limited by the resolution of the voxels (which is often limited by computation time and available memory). This limitation is illustrated in figure 1.16(*a*).

To overcome this limitation a parametrized surface representation may be used instead of voxel occupancy. In [10] a parametrized surface fit was obtained using a cost function based on image correlation. Here we extend the cost function to also enforce the geometric constraints arising from the apparent contour.



(a) (b)

Figure 1.16 The apparent contour as projected from the generated volumetric model of the visual hull. (a) A low resolution model has been chosen to exaggerate the re-projection errors. This model has approximately 8000 polygons and gives an approximate reprojection error of 5–7 pixels. (b) The errors are reduced to sub-pixel if the cost function of equation (1.2) is used in a minimization over the surface.

Intensity correlation. The cost function used in [10] has the form

$$\mathcal{C}_t = \frac{1}{4pq}\int_{-p}^{+p}\int_{-q}^{+q}\left(I_1(m_1+m)-\bar{I}_1(m_1)\right)\left(I_2(K(m_1+m))-\bar{I}_2(K(m_1)\right)dm \ , \tag{1.1}$$

thus correlating a texture patch, centred around image point m_1 in a first image, $I_1(m)$ with the surface-mapped patch in a second image, $I_2(m)$. The function $K(m)$ maps points between the two images given a hypothesized surface shape (the optimization parameter), S. The mean patch intensities, $\bar{I}_1$ and $\bar{I}_2$, allow for an affine scaling of intensities.

Here, we extend the cost function by introducing an extra distance term to minimise errors in the reprojected surface:

$$\mathcal{C}_a = \int d(\mathbf{P}(S)(s), \mathbf{c}(s))ds \ . \tag{1.2}$$

The distance function, d, measures the distance between the projected apparent contour, $\mathbf{P}(S)$ from the apparent contour detected in the original images, $\mathbf{c}$.

The overall cost function balances these two terms according to variance of the noise probability distributions, σ_t (of intensity cross-correlation) and σ_a (of pixel localization error), on each one of the cost terms:

$$\mathcal{C} = \frac{\mathcal{C}_t}{\sigma_t^2} + \frac{\mathcal{C}_a}{\sigma_a^2} \ . \tag{1.3}$$

The importance of including this term is that the surface is now constrained (by the apparent contour) at regions at which there is no texture. This information is unused in the original work [10] because the cost is not included.

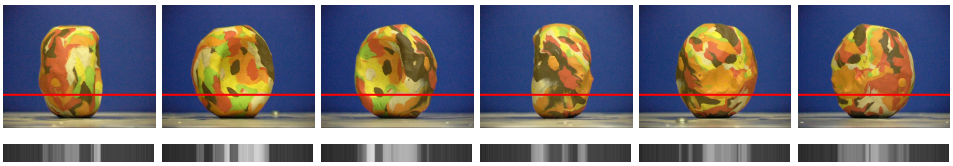


Figure 1.17 Six images (*top*) from a sequence of 100 images of a plasticine object with a single large concavity. The plane common to the cameras is intersected with each image plane and the 1D intensity image corresponding to the line of intersection (*bottom*) taken as an example in section 4.1.

4.1 TWO DIMENSIONAL EXAMPLE—IMPLEMENTATION

Figure 1.17(*top*) shows a turn-table image sequence of a plasticine object with a large concavity. As an example, we will consider a cross-section of the object by considering the images along a set of corresponding epipolar lines (figure 1.17(*bottom*)).

The visual hull is used as the starting point for the optimization (computed using the algorithm outlined in section 1.8). By applying a similar indicator function to that used in section 3.3, the correctly reconstructed regions of the curve are located and held constant during the optimization (figure 1.19(*a*)).

For the purposes of this example (see discussion below), the curve is represented as a piecewise linear spline, with approximately 150 line segments. The vertices are restricted to move along the curve normals. The cost function (1.3) is minimized by moving the vertex positions of the curve using a Levenberg Marquardt algorithm. A Z-buffer is maintained during the optimization to compute the reprojected apparent contour, and to ensure that occluded regions are not included in the correlations. Figure 1.19(*b*) show the results of this optimization if the texture term alone is used (it is clear that the algorithm has only converged to a local minimum, and is far from the correct solution), whilst figure 1.19(*c*) shows how the apparent contour term improves the model considerably. In the concave region, results (*b*) and (*c*) are identical as the apparent contour provides no information. It is clear that the texture does not provide sufficient information alone (indeed, figure 1.19(*d*) violates the visual hull constraint) and is augmented by the apparent contour constraint in the convex regions.

The gourd sequence of figure 1.15 is used as a 3D example using the cost function of equation 1.2. Figure 1.16 demonstrates how the reprojection errors of the apparent contour reduce from 5–7 pixels to subpixel.

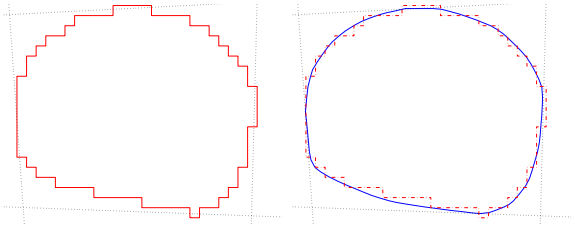


Figure 1.18 (a) A voxel model of the visual hull. Due to sampling errors, it does not accurately obey the apparent contour constraints. (b) Applying the cost function of equation (1.2) (minimizing errors in the reprojected apparent contour along), the model is significantly improved from the voxelated visual hull.

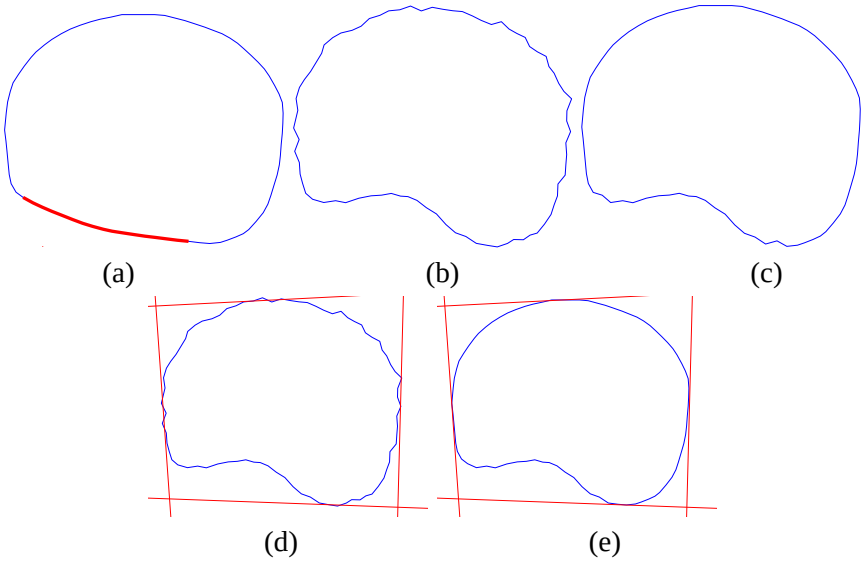


Figure 1.19 (*a*) The cost function (1.3) is applied to the surface of the visual hull. Areas with high cost (incorrectly modeled regions such as the concavity) are marked. (*b*) A model generated by minimising the cost function of equation (1.1) is shown. The concavity is now more accurately modeled, but if the full cost function (equation (1.3)) is used the convex regions are further constrained by the apparent contour constraint (*c*). (*d*) and (*e*) show the results of (*b*) and (*c*) with two of the back-projected apparent contours superimposed. (*d*) clearly shows that the model does not obey the apparent contour constraint if equation (1.2) is not directly included in the cost function (as is the case in (*e*)).

5. SUMMARY

The algorithm presented here enables an accurate and complete model of a surface to be built automatically from multiple images, and we now discuss limitations and possible extensions to this approach.

Apparent contours. This contribution to the cost function involves a geometric image error minimization. The error makes minimal assumptions about the world (cf the photo-consistency constraint discussed below). Its use requires that the apparent contour is correctly segmented from the image (here this is achieved by 'blue-screening') and this is the point at which errors can occur There are two failure modes: first, under segmentation (where parts of the contour are missed). This failure is not particularly serious because the surface can still be carved away from other views where the contour is detected; the second type of failure, over segmentation, is serious because parts of the actual object will then be removed.

Photo-consistency. This requires assumptions about the photo-metric properties of the surface, i.e. about the BRDF, which is generally assumed to be Lambertian. A serious problem is with specularities where the assumed properties are strongly violated. The naive use of the space carving algorithm is inappropriate in such cases. Unfortunately, specularities do commonly occur, e.g. for metallic and shiny surfaces. However, it is possible to verify the consistency of the BRDF assumptions: at the epipolar net points the visual hull generates surface points (and the surface normal), and the brightness of these points can be used to assess, for example, the Lambertian assumption.

Extensions. The presence of specularities can also be turned to an advantage if the position of the light source is known. In this case, based on the standard geometry of mirror reflection, the surface (position and normal) can be reconstructed up to a one parameter family. Shadows may also be used to provide geometric constraints.

In the current implementation, it has been assumed that the visual hull correctly captures the topology, and the topology does not subsequently change in the optimization stage. We are currently investigating other parameterizations of the surface, such as the level sets used in [10]) and the T-snakes developed in [23] which will enable changes in surface topology within the optimization.

Acknowledgements

Figure 1.2 was provided by Roberto Cipolla and Peter Giblin. The dinosaur image sequence of figure 1.1 was provided by the University of Hannover. The gourd sequence (figure 1.15) was provided by Edmond Boyer of INRIA, Grenoble. Funding for this work was provided by the Engineering Physical Sciences Research Council (EPSRC).

References

[1] N. Ayache and O. D. Faugeras. Building a consistent 3D representation of a mobile robot environment by combining multiple stereo views. In *Proc. International Joint Conference on Artificial Intelligence*, pages 808–810, 1987.

[2] H. H. Baker and T. O. Binford. Depth from edge and intensity based stereo. In *IJCAI81*, pages 631–636, 1981.

[3] P. Beardsley, P. Torr, and A. Zisserman. 3D model acquisition from extended image sequences. In *Proc. European Conference on Computer Vision*, LNCS 1064/1065, pages 683–695. Springer-Verlag, 1996.

[4] R. Berthilsson and K. Åström. Reconstruction of 3-D curves from 2-D images using affine shape methods for curves. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 1997.

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

[6] R. Cipolla, K. Åström, and P. Giblin. Motion from the frontier of curved surfaces. In *Proc. 5th International Conference on Computer Vision, Boston*, pages 269–275, 1995.

[7] R. Cipolla and A. Blake. Surface shape from the deformation of apparent contours. *International Journal of Computer Vision*, 9(2):83–112, 1992.

[8] G. Cross, A. W. Fitzgibbon, and A. Zisserman. Parallax geometry of smooth surfaces in multiple views. In *Proc. 7th International Conference on Computer Vision, Kerkyra, Greece*, pages 323–329, September 1999.

[9] G. Cross and A. Zisserman. Quadric surface reconstruction from dual-space geometry. In *Proc. 6th International Conference on Computer Vision, Bombay, India*, pages 25–31, January 1998.

[10] O. Faugeras and R. Keriven. Complete dense stereovision using level set methods. In *Proc. 5th European Conference on Computer Vision, Freiburg, Germany*, pages 379–393, 1998.

[11] A. W. Fitzgibbon, G. Cross, and A. Zisserman. Automatic 3D model construction for turn-table sequences. In R. Koch and L. Van Gool, editors, *3D Structure from Multiple Images of Large-Scale Environments, LNCS 1506*, pages 155–170. Springer-Verlag, June 1998.

[12] A. W. Fitzgibbon and A. Zisserman. Automatic camera recovery for closed or open image sequences. In *Proc. European Conference on Computer Vision*, pages 311–326. Springer-Verlag, June 1998.

[13] P. Giblin and R. Weiss. Reconstruction of surfaces from profiles. In *Proc. 1st International Conference on Computer Vision, London*, pages 136–144, London, 1987.

[14] W. E. L. Grimson. *From Images to Surfaces: A Computational Study of the Human Early Visual System*. MIT Press, 1981.

[15] T. Joshi, N. Ahuja, and J. Ponce. Structure and motion estimation from dynamic silhouettes under perspective projection. In *Proc. 5th International Conference on Computer Vision, Boston*, pages 290–295, 1995.

[16] R. Koch. 3D surface reconstruction from stereoscopic image sequences. In *Proc. 5th International Conference on Computer Vision, Boston*, pages 109–114, 1995.

[17] J. J. Koenderink. What does the occluding contour tell us about solid shape? *Perception*, 13:321–330, 1984.

[18] K. Kutulakos. Affine surface reconstruction by purposive viewpoint control. In *Proc. 5th International Conference on Computer Vision, Boston*, pages 894–901, 1995.

[19] K. Kutulakos and S. Seitz. What do N photographs tell us about 3D shape? Technical Report 680, University of Rochester, January 1998.

[20] K. Kutulakos and S. Seitz. A theory of shape by space carving. In *Proc. 7th International Conference on Computer Vision, Kerkyra, Greece*, pages 307–314, 1999.

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

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

[23] T. McInerney and D. Terzopoulos. Topologically adaptable snakes. In *Proc. 5th International Conference on Computer Vision, Boston*, pages 840–845, 1995.

[24] S. B. Pollard, J. E. W. Mayhew, and J. P. Frisby. PMF: A stereo correspondence algorithm using a disparity gradient limit. *Perception*, 14:449–470, 1985.

[25] J. Porrill and S. B. Pollard. Curve matching and stereo calibration. *Image and Vision Computing*, 9(1):45–50, 1991.

[26] J. H. Rieger. Three dimensional motion from fixed points of a deforming profile curve. *Optics Letters*, 9(1):123–125, 1986.

[27] S.M. Seitz and C.R. Dyer. Photorealistic scene reconstruction by voxel coloring. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition, Puerto Rico*, pages 1067–1073, 1997.

[28] J. A. Sethian. *Level Set Methods.* Cambridge University Press, Cambridge, 1998.

[29] A. Shashua and S. Toelg. The quadric reference surface: Theory and applications. *Proc. International Conference on Computer Vision*, 1997.

[30] S. Sullivan and J. Ponce. Automatic model construction and pose estimation from photographs using triangular splines. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(10):1091–1096, October 1998.

[31] R. Szeliski. Rapid octree construction from image sequences. *CVGIP*, 58(1):23–32, July 1993.

[32] R. Szeliski and R. Weiss. Robust shape recovery from occluding contours using a linear smoother. *International Journal of Computer Vision*, 28(1):27–44, June 1998.

[33] R. Vaillant and O. D. Faugeras. Using extremal boundaries for 3-D object modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):157–173, February 1992.

[34] Z. Zhang, R. Deriche, O. D. Faugeras, and Q. Luong. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. *Artificial Intelligence*, 78:87–119, 1995.