Topical Editor: Prof. Dr. Eun-Soo Kim, 3D Res. **03**, 04(2012)6
10.1007/3DRes.04(2012)6

3DR EXPRESS

3D surface reconstruction from multiview photographic images using 2D edge contours

Simant Prakoonwit • Ralph Benjamin

Received: 17 January 2012 / Revised: 11 May 2012 / Accepted: 20 June 2012

Abstract Most techniques for reconstructing 3D shapes from multi-view 2D photographic images require a large number of images. In this paper, we present a new method for reconstructing 3D surfaces, represented by sets of polygons, using a small number, e.g. 10, of 2D photographic images with full prior knowledge of camera configurations. The method is automatic. Unlike most currently available silhouette-based multiview reconstruction methods, 3D surface points and surfaces are reconstructed directly from 2D edges without costly intermediate voxel reconstruction. The surfaces reconstructed by the proposed method are self-optimized. More surface points and polygons are automatically generated on highly curved parts of a surface. Experiments on computer generated objects and real physical objects were conducted to verify the method.

Keywords 3D reconstruction, apparent contour, contour generator, epipolar, multiview, space curve, surface point, wireframe, surface reconstruction

1. Introduction

1.1 3D reconstructions from multiview images

3D object reconstruction from multiple 2D images is an important problem in computer vision with many applications. Shape-from-silhouette is one of the widely known 3D reconstruction methods[1-4]. Volumetric reconstruction and representation are used in shape-from-silhouette. The 3D shape of an object is computed from the intersection of the silhouette cones, which are back-projected from all multiview. Martin and Aggarwal[5] were the first to propose this method. Octrees representation[6] is wildly used for this type of reconstruction. The approach was subsequently investigated by many researchers[7-11]. The major advantage of volumetric approaches is there robustness in reconstructing objects with complicated topology. However, the methods require a large number of images and generate a huge number of vexes to be manipulated. For example, the method proposed by Kolev and Brox et al.[4] generates more than 20 million voxels and requires parallel computing to implement the algorithms. If the surface of an object is required, then a costly process has to be employed to extract the surfaces from the volumetric data[3, 12].

3D surface models can also be reconstructed directly from few silhouettes by using statistical or parametric models such as the methods described[13-16]. However, the information of the shape to be reconstructed has to be known in advance and the methods cannot reconstruct any other shapes.

In 3D reconstruction, more information can be gained by not only using silhouette or occluding apparent contours, but all 2D apparent contours[17-19]. Many methods have been proposed to recover a 3D surface from its apparent contours on multiview images[17, 20, 21]. Cipolla and Blake[18] demonstrated that the curvature and depth of the contour generator points can be recovered, provided that the camera motion is known. Edge contours or apparent contours from images taken from a circular motion around an object were also used[22] to reconstruct the object. A new method was, in addition, proposed[22] to incorporate additional views to improve both the shape and textures of the reconstructed 3D object. However the authors did not mention the criteria for

Simant Prakoonwit[1] (✉) • Ralph Benjamin[2]
[1]Department of Computer Science and Technology, University of Bedfordshire, Bedfordshire, LU1 3JU, UK,
Tel: +44 77 1315 9433
E-mail: Simant.Prakoonwit@ieee.org, Simant.Prakoonwit@beds.ac.uk

choosing the addition views. It is likely that the additional viewing directions depend on the shape of the object.

1.2 Proposed contribution

In this paper, we propose an effective automatic method to reconstruct full surfaces of 3D objects using epipolar geometry and all the corresponding edges from multiview 2D photographic images. Overall, our proposed reconstruction method does not assume that objects come from some classes such as surfaces of revolution, algebraic surfaces or generalized cylinders. Compared to other existing shape-from-silhouette methods, our main contribution can be summarized as follows:

The proposed method requires a smaller number, e.g. 10, of images compared to 20-36 images needed[4], 40 images[23], 42 images[21], and 280 images used for reconstructing HOT curves[24].

Only edges are needed for the reconstruction process, unlike most existing techniques[3, 4], where the additional information from images is required. Hence the proposed method should be useful when an object of interest is textureless, lighting configurations are not available or shadows are not reliable, etc.

The method reconstructs surfaces directly from the edges in images without costly intermediate volumetric reconstruction[3, 4] as mentioned in the Section 1.1.

The characteristics of the wireframes of intersection points generated by our method are quite unique. Existing surface-from-wireframe algorithms[25-27] cannot be employed. Hence we present a more flexible surface-from-wireframe in this paper. The details are discussed in Section 2.3.1.

Objects reconstructed by the proposed method are self-optimized automatically. There are more surface points and polygons on a highly curved part and less surface points and polygons on a smooth part.

The work is based on our previous work[28] and a preliminary version of this framework presented at a conference[29]. The method presented in the first paper can only reconstruct surface points and wireframes. In the conference paper, a simplified version for full surface reconstruction was described. In the following, we summarize the novelty of this paper over our two previous papers.

The accuracy of the reconstruction method mostly depends on correct edges in images. Unlike our previous work[28-29] where a simple edge detection process was used, we apply multi-scale analysis over scale-map, automatic Canny edge detection and edge segment linking process incorporating Gestalt laws to detect and identify edges. Then the Minimum Description Length and Potential for Energy-Reduction Maximization were applied to accurately represent edges using B-spline curves. This helps eliminating incorrect and jagged edges appeared in the previous results.

We have developed an improved version of the preliminary surface reconstruction from wireframe method[29] by using secondary surface points combining with the surface normal unification method. The improvement can be clearly seen when comparing the results in this paper with the rough results presented in our conference paper[29].

2. The approach

2.1 3D surface point reconstruction

In this section, we briefly explain some geometric fundamentals and our method of pairing multiview profiles[28] to reconstruct 3D frontier points and the network of contour generators of objects.

In multiview images where an object is projected or seen from different viewpoints, contour generators generated from different projections form a wireframe representing the object's surface, as an example shown in Fig. 1. The methods to reconstruct the wireframe and to create full surface representation of an object from the wireframe are presented.

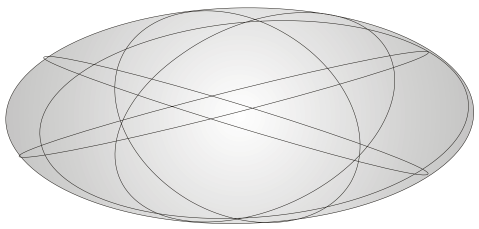


Fig.1 A network of contour generators from different projection forms a wireframe on an ellipsoid's surface.

2.1.1 Epipolar geometry, apparent contours and frontier points

In this section some principles and terminologies used in this paper are described. Assume there are two pinhole photographic cameras centered at $\mathbf{X}_1$ and $\mathbf{X}_2$. S, a surface of type C^1, is projected onto two image $\mathbf{I}^1$ and $\mathbf{I}^2$ as shown Fig. 2. The following definitions can be applied[30-31]:

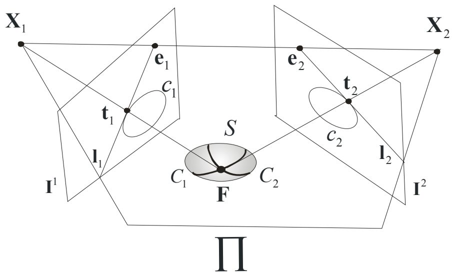


Fig. 2 A 3D surface is projected onto two image planes by two cameras centered at $\mathbf{X}_1$ and $\mathbf{X}_2$.

Space curve $C_1 \subset S$ is a curve in 3D space. All the points $\mathbf{C} \in C_1$ are the tangent points where the ray to camera optical centre $\mathbf{X}_1$ is tangent to S at $\mathbf{C}$. The space curve is called a *contour generator*. The same principle is also applied to space curve $C_2 \subset S$ associated with camera optical center $\mathbf{X}_2$. The projection of the contour generator, where the camera centre is the centre of projection, onto the corresponding image plane is called *edge* contour or *apparent contour*, e.g. c_1, c_2 in Fig. 2.

An *epipolar plane* Π is the plane that contains the *epipolar baseline* $\mathbf{X}_1\mathbf{X}_2$. The epipolar baseline intersects the two image planes at *epipoles* $\mathbf{e}_1$ and $\mathbf{e}_2$. The point where an epipolar plane Π is tangent to surface S and where two contour generators C_1 and C_2 intersect is called a *frontier point*, denoted by $\mathbf{F}$. The apparent contours c_1 and c_2 are tangent to the *epipolar tangent lines* $\mathbf{l}_1$ and $\mathbf{l}_2$ at *epipolar tangent points* $\mathbf{t}_1$ and $\mathbf{t}_2$, respectively, in the two image planes. In other words, the epipolar tangent points are the projections onto the images of the same point on S, namely frontier point $\mathbf{F}$.

The network of reconstructed frontier points and contour generators from multiview projected images provides a wireframe containing information on the surface of an object of interest.

2.1.2 Apparent contour detection and representation

Each image is enhanced to reduce the noise level, eliminate unwanted edges, and sharpen edges and to increase the contrast of the edges. Then an automatic edge detection method is applied to extract apparent contours form images. In this work, all apparent contours have to be represented analytically. Therefore, each contour is fitted with cubic B-spline curves.

a) Edge detection and apparent contour extraction

We have adopted and modified the methods presented[32, 33] to automatically detect edges and extract apparent contours from images. The scheme can be explained as follows:

1) Multiscale analysis over scale-map. An image is iteratively simplified (or cartoonized) using a bilateral filter[34]. The scales are emerged from the iterations of bilateral filtering and represent successively simplified versions of the original image. The process aims to eliminate unwanted details from texture, noise, illumination effects, etc. in images. Those details mainly create edges which are not apparent contours.

2) Automatic Canny edge detection. At each scale, Canny edge detection is applied on the image filtered by the bilateral filter. All edges in all images across all scales are used to determine the edges most likely to represent objects' boundaries and apparent contours[33]. As we aim to make the whole process automatic, we propose that the parameters of Canny edge detection should be found by using the method described[32]. This method uses a saliency map[35] to incorporate spatial local organization considerations to purely statistical correspondence considerations.

3) Edge segment linking. All the edge segments are analyzed and then linked up to form object boundaries and other meaningful edges by using the techniques described[33]. Edge analysis and linking are based on minimizing a cost function created according to Gestalt laws[36]. The laws describe the way our mind group component parts to form whole objects by using eight visual cues: *proximity, similarity, good continuation, closure, common fate, surroundedness, relative size,* and *symmetry*. Some of the processed edges will be used as apparent contours and the rest will be automatically filtered out in the subsequent processes.

b) Edge representation

In this work, we represent edges and apparent contours analytically. Each edge or apparent contour is fitted with cubic B-spline curves. This method allows the localization of contour points with sub-pixel precision, which may improve the accuracy of the reconstructed 3D points especially when the object is relatively small in the image. To conduct B-spline curve fitting automatically, we adopt the technique described[37]. The minimum description length (MDL) principle and a control point insertion strategy based on maximizing the Potential for Energy-Reduction Maximization (PERM) are combined to produce an automatic and reliable scheme for spline-fitting. Note that edges which are not apparent contours are normally filtered out during the epipolar matching process.

2.1.3 3D frontier points reconstruction by paring different views

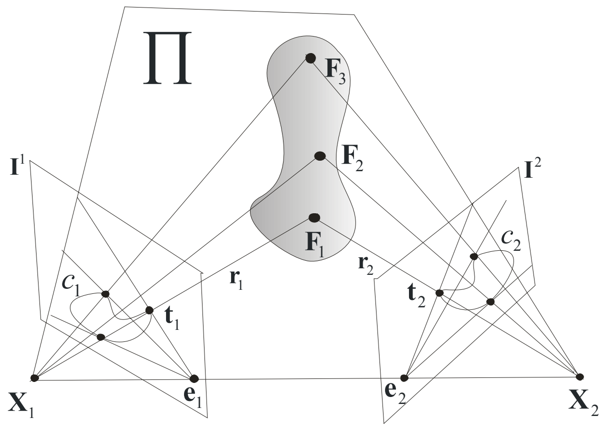


Fig. 3 Epipolar tangent lines, epipolar tangent points and frontier points can be created by imagining Epipolar plane Π rotating about the epipolar baseline X_1 and X_2 touching the surface at different points.

In this section, based on our previous work[28], we describe how multiple 3D frontier points on a 3D surface can be found by pairing 2D images, using epipolar geometry in a novel way. In each pair of 2D images, a frontier point can be found from the intersection of a pair of rays from the two camera centers which pass through epipolar tangent points on corresponding apparent contours. For example, consider an object S in Fig. 3, an apparent contour generated from S is not a simple convex curve[30]. Frontier point $\mathbf{F}_1$ can be determined by the intersection of rays $\mathbf{r}_1$ and $\mathbf{r}_2$. To find more frontier points, one can imagine the epipolar plane rotating about epipolar base line $\mathbf{X}_1\mathbf{X}_2$. As the plane rotating, it is tangential to the object's surface at different points, more frontier points are found, e.g. $\mathbf{F}_2$, $\mathbf{F}_3$. The more complex the surface, the greater is the number of 3D points required, but the greater the number of 3D frontier points generated.

In case of multiple views, the same principle can be applied. Consider a system of N_{pl} cameras, viewing surface S from different directions generating N_{pl} 2D images. From this set, we will have $N_{pl}(N_{pl}-1)/2$ image. If surface S is convex and closed, then $N_{pl}(N_{pl}-1)$ frontier points will be found. A nonconvex object will automatically generate more frontier points from the pairing.

2.2 Reconstruction of contour generator networks

The frontier points on each contour generator can be reconstructed by pairing all the 2D images. 3D contour generator can then be defined by linking the frontier points, in the same sequence as their corresponding tangent points on the apparent contour[28]. If reconstructed contour generators have to be smooth curves, then cubic splines[38] can be used to interpolate the frontier points. The same process can be applied to reconstruct other all contour generators from other pairs of images. As a result, the network of contour generators or wireframe representing the surface of the object can be acquired.

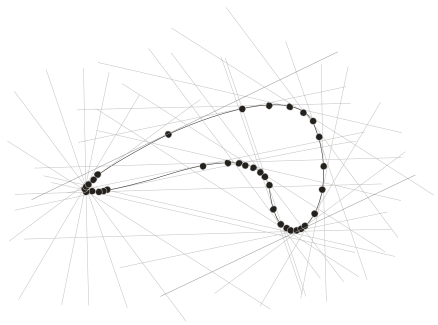


Fig. 4 Approximate distribution of epipolar tangent points and epipolar tangent lines along an apparent contour. Automatically, there are more points on high curvature parts of the contour.

The more 2D images the more epipolar tangent lines and tangent points, as illustrated in Fig. 4. In an ideal convex case, if our viewing angles are uniformly distributed in solid angle, smooth shapes, with a monotonic change of gradient, result in tangent points at approximately equal increments of tangent angle. In a ten-view system, on each 2D apparent contour, pairing with nine other views will establish 18 epipolar tangent points, each centered on an arch of ±10 °, for a total of 360 °. A sharp corner will generate multiple coincidental additional epipolar tangent points. As the points thus defined on the objects will be evenly spread, in tangent angles, they cluster closely in highly curved regions and are widely spread in smooth, flat parts of the surface; the optimum distribution.

To increase the quality of reconstructed 3D contour generators, more 3D points, called secondary points, can be estimated and added between two consecutive frontier points. The frontier points are reconstructed in such a way that, each segment of a contour generator between two consecutive frontier points can be roughly approximated as a plane curve. The method of adding secondary points is illustrated in Fig. 5, where three secondary vertices $\mathbf{F}_3$, $\mathbf{F}_4$ and $\mathbf{F}_5$ are estimated between frontier points $\mathbf{F}_1$ and $\mathbf{F}_2$. On apparent contour c, Points $\mathbf{t}_3$, $\mathbf{t}_4$ and $\mathbf{t}_5$ are determined at equal space in terms of tangent angles so that the points are cluster closely around high curvature parts of the contour. Plane $\mathbf{I}'$ is perpendicular to plane $\mathbf{F}_1\mathbf{t}_1\mathbf{t}_2\mathbf{F}_2$. Then the 3D points $\mathbf{F}_3$, $\mathbf{F}_4$ and $\mathbf{F}_5$ are the intersections of plane $\mathbf{I}'$ and the rays from the corresponding camera centre $\mathbf{X}$ to $\mathbf{t}_3$, $\mathbf{t}_4$ and $\mathbf{t}_5$ respectively.

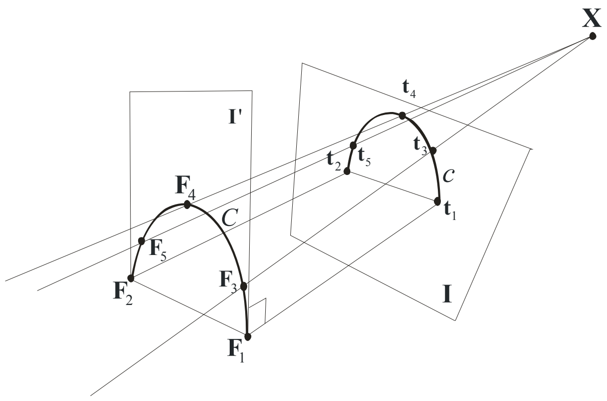


Fig. 5 Additional 3D secondary points $\mathbf{F}_3$, $\mathbf{F}_4$ and $\mathbf{F}_5$ on contour generator C can be estimated from additional 2D points $\mathbf{t}_3$, $\mathbf{t}_4$ and $\mathbf{t}_5$ on apparent contour c.

2.3 Surface reconstruction

From the last section, a network of contour generators is reconstructed. However, this is a wire-frame representation of the original object. The complete surface information is not presented. We have selected the B-rep (Boundary representation) scheme to represent the full surfaces of the reconstructed objects. Each reconstructed object is represented by a set of surface patches or polygons. Each patch or polygon consists of a sequence of vertices, or frontier points and secondary points.

2.3.1 Surface reconstruction from wireframe

Many researchers have developed methods for turning wireframe objects into full surfaces or solid models[25-27]. However, in most proposed methods, the wireframes have to be well defined and have to belong to surfaces of some certain types.

The reconstructed contour generator networks or wireframes from our method have different characteristics in terms of contour generator connections and the way frontier points are distributed. The frontier points are not evenly spaced. Some contour generators are discontinuous. Therefore usual existing surface reconstruction from wireframe algorithms often cannot produce acceptable results.

Chen et al.[39] proposed a new method to reconstruct full surfaces from more complicated wireframes. In this method, evenly spaced parallel cross-sectional planes are used to re-sample a wireframe model. On each cross-sectional plane, the intersection points between the wireframe and the plane are then linked up to form the 2D cross-sectional boundary contours of the object. The boundary contours on all planes are joined to form a complete surface model. However, this method is still not suitable and effective for the wireframes reconstructed by our approach. The optimal distribution of our 3D frontier points will be eliminated by the wireframe re-sampling. Moreover, a large number of unnecessary 3D frontier points and polygonal faces will be created. Also, joining up points on each cross-sectional plane to form correct boundary contours and linking up the contours on different planes correctly are difficult tasks to perform.

2.3.2 Incremental surface reconstruction using locally projected wireframe

In this article, the authors propose a reconstruction method which uses a locally projected wireframe or contour generator network to reconstruct surface polygons incrementally. At each polygon to be reconstructed, a set of local frontier points and wireframe segments linking the frontier points are projected onto an appropriate plane. This is to convert 3D polygon identification problem into a 2D problem which is much more straightforward to be solved using a minimum angle criterion. The process is repeated until the full set of polygons representing the surface is found. Note that only reconstructed frontier points are considered first. Reconstructed secondary vertices will be taken into account later in the triangulation process. The process can be described in detail as follows:

Step 1. Local projection plane determination

To find a polygonal face of a reconstructed wireframe correctly, we suggest to locally flattening the wireframe around the face, so that the sequence of the frontier points of the polygon can be identified in 2D. This can be done by finding a suitable local projection plane. Then all relevant frontier points and wireframe segments or edges are projected onto the plane. The projections must form a planar graph. There must be no intersection of the projected wireframe segments on the plane, apart from the original intersections at frontier points.

It is not straightforward to create such a plane. We propose a method to adjust the local projection plane to create the best locally flattened wireframe. To illustrate the method, consider an example in Fig. 6. Suppose $\mathbf{F}_2$ is selected as the first seed frontier point and P1 is the first polygon to be identified in the wireframe. The neighboring frontier points in 'connection layer 1' or the frontier points which are directly connected to the seed point $\mathbf{F}_2$ are selected, i.e. $\mathbf{F}_8$, $\mathbf{F}_{18}$, $\mathbf{F}_{14}$, see Fig. 6. Local projection plane **P** is then determined by using a standard least square fitting technique[40] to fit the seed points and the neighboring points.

Once the local projection plane is defined, the frontier points and the wireframe segments, or edges, connecting those vertices, i.e. E_2, E_3, E_{40}, are projected onto the plane. Those edges are labeled as connection layer 1 edges. If the projected edges and frontier points form a planar graph on the local projection plane, then we move to step 2. If not, then the frontier points in connection layer 2 or the frontier points that are connected to the vertices in connection layer 1, are added to form a new set of frontier points. To check if the projected edges and frontier points form a planar graph, we test if there is an intersection of the edges, apart from the original intersections at the frontier points. If there is no intersection, we can assume that the graph is planar. The updated local projection plane is found by using the least square fitting technique to fit the new set of frontier points from connection layers 1 and 2. The aim is to find a local projection plane which contains all the frontier points or as many frontier points as possible of polygon P1. From experiments, normally the maximum number of layers required is 3, but, in general, setting the maximum number of layers to 2 is enough to generate acceptable results in most cases.

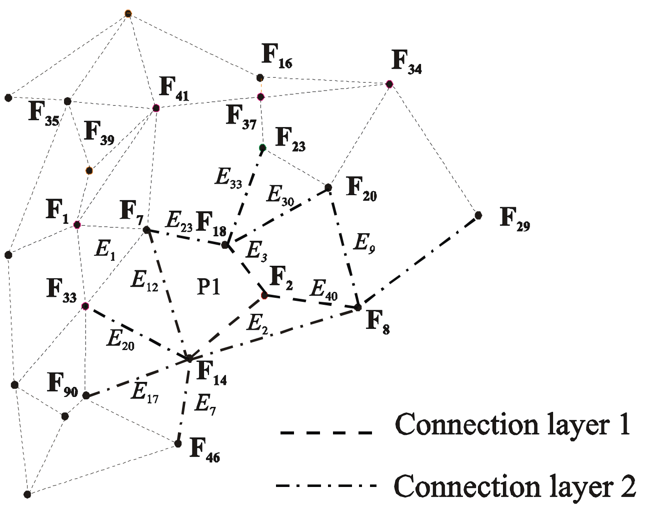


Fig. 6 Local network of contour generators used to determine polygonal P1.

If there is an intersection in layer 2 or higher, then the local projection plane defined from the vertices in the previous layer is used. However, in a very rare exceptional case where there is an intersection of edges on the local projection plane defined by connection layer 1 vertices, a new seed point is randomly selected from other vertices.

Step 2. Polygon identification using the minimum angle criterion

Once an appropriate local projection plane is found, all projected edges and vertices are used to determine the polygon face defined by a sequence of vertices. For example, if the vertices in connection layers 1 and 2 in Fig. 6 are used to create the local projection plane **P**, Fig. 7, where $\mathbf{f}_n$ is the projection of $\mathbf{F}_n$ (n is a frontier point number) onto the local projection plane **P**. The problem is now transformed into polygon identification in 2D coordinates which can be solved by using the standard minimum interior angle method. The frontier point sequence of the patch is therefore found by using the minimum angle, starting from e_2 (projection of E_2), as the criterion for choosing the next point in the sequence, Fig. 8. $\theta_1, \theta_2, \theta_3$ are the minimum angles. In this case the minimum angle criterion is in clockwise direction. Polygon P1 can therefore be defined by the sequence of frontier points $\mathbf{F}_{14}$, $\mathbf{F}_2$, $\mathbf{F}_{18}$ and $\mathbf{F}_7$ consecutively. Each surface polygon must also satisfy 2-manifold orientable surface conditions[41, 42]. For each newly reconstructed polygon, it must not intersect with any of the polygons previously identified, except along the edges shared with other polygons. If there is an intersection, the search for the next face of polygon will be conducted in the other direction. For example, in Fig. 8, if P1 intersects with an existing polygon, the search from e_2 and $\mathbf{f}_2$ turn to the other direction by using the minimum angle criterion in anti-clockwise direction. Each edge can be shared by only two polygons.

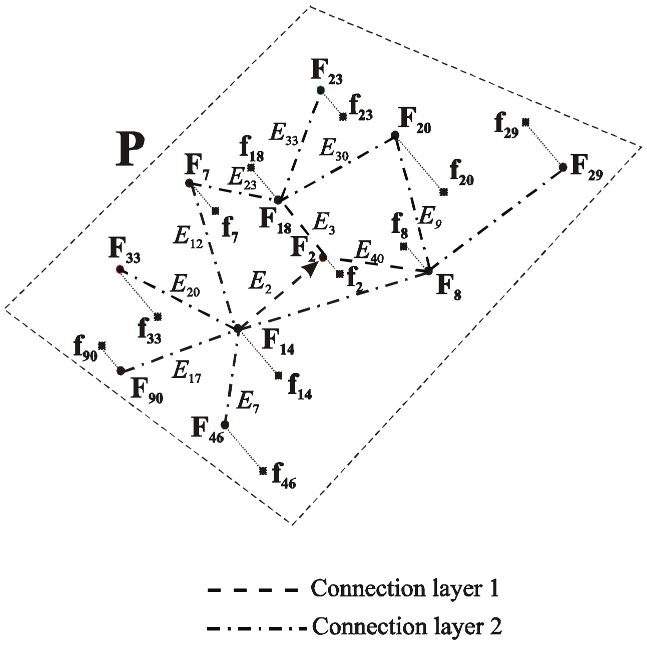


Fig. 7 To find the surface polygon that consists of $\mathbf{F}_2$, $\mathbf{F}_{18}$, $\mathbf{F}_7$ and $\mathbf{F}_{14}$, the seed point and the neighbouring points are projected onto local projection plane **P.**

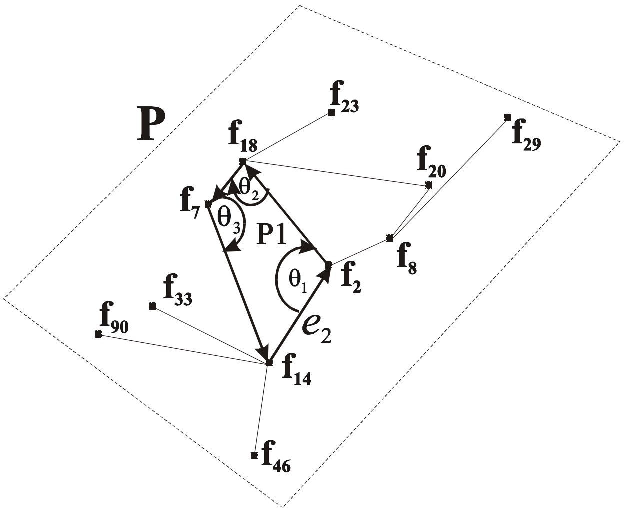


Fig. 8 The sequence of frontier points in the polygon is determined by using minimum angles as the criterion for choosing the next point in the sequence.

Note that this method works only if the projected edges and frontier points on the local projection plane form a planar graph. This condition, therefore, is used to check and adjust the local projection plane when a new polygon vertex is found. The new vertex is used as a new seed point to find additional frontier points to adjust the local projection plane as described in Step 1.

For the next polygon, the new seed frontier point and the starting edge are selected from the newly found polygon. Then Steps 1 and 2 are repeated until all the polygons are found.

Step 3. Triangulation of polygonal faces

After all polygons are identified, secondary vertices are added to each polygon, Fig. 9(a). Then each polygon is triangulated to produce a set of triangular facets, Fig. 9(b). The triangulation process is based on the standard Ear Clipping algorithm[43]. An 'ear' of a polygon is a triangle created by three consecutive vertices for which no other vertices of the polygon are inside the triangle and has two sides on the edge of the polygon. In our case, the ear of a polygon is determined by the shortest 'diagonal' of the polygon. Once an ear is found, it is removed from the polygon resulting in a new polygon. The process is repeated until there is only one triangle left.

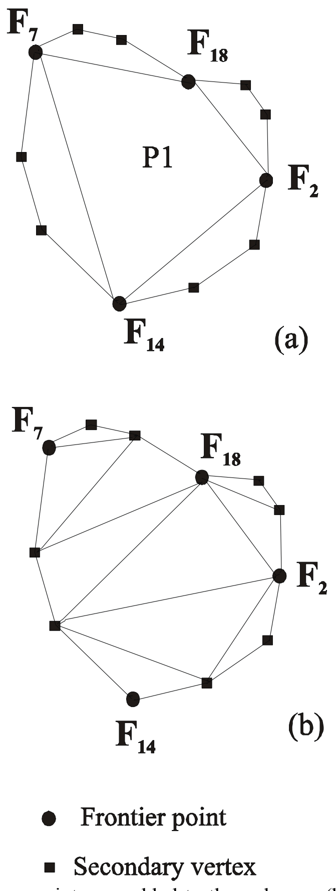


Fig. 9 (a) Secondary points are added to the polygon. (b) The polygon is triangulated to create a set of triangular facets.

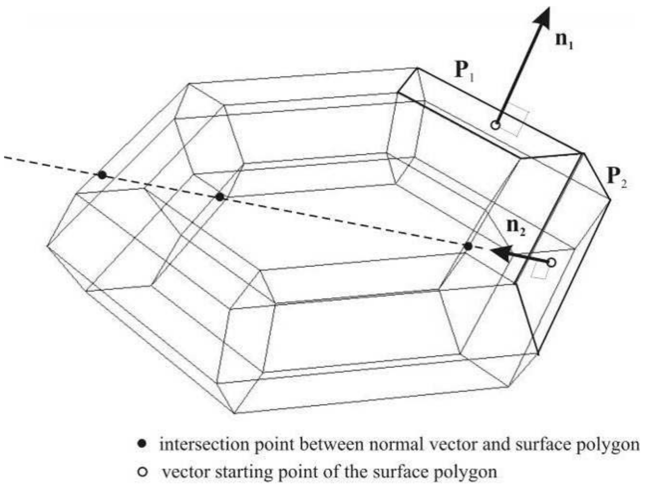
Step 4. Surface normal unification

The processes in Steps 1 to 3 do not generate unified triangular facets' normals. This will cause problems when smoothing and rendering the surface. Some triangular facets may have their normal vectors pointing outwards from the object, while some may have theirs pointing inwards. To unify all the triangular facets' normals, we aim to set all the vectors to point outwards.

We assume that the reconstruction process creates an approximately closed surface. Therefore, the direction of a normal vector can be tested by counting the number of intersections between the normal vector line and the surface. A normal vector line is a line starting from a point on the corresponding triangular facet and extending in the direction of the normal vector. If the number of intersections is zero or an even number, not including any tangential points, then the normal vector is pointing outwards. If the number of intersections is an odd number, then the normal vector is pointing inwards. An example is shown in Fig. 10. Note that the surface in the figure is represented by a set of planar polygons instead of a set of triangular facets. This is just to simplify the figure for illustration purposes. In the example, normal vector $\mathbf{n}_1$ of polygon $\mathbf{P}_1$ is pointing outwards and the number of the intersections is equal to zero. While the number of intersections between the normal vector line of $\mathbf{n}_2$ of polygon $\mathbf{P}_2$ is equal to three which is an odd number. Therefore $\mathbf{n}_2$ is pointing inwards. Once an inwardly pointing normal vector is detected, it can be straightforwardly flipped to point outwards. All reconstructed triangular facets are checked in this manner to create a unify set of normal vectors across an object's surface.

2.4 Viewing directions

We use viewing directions which are uniformly distributed over a hemisphere. The viewing directions can be derived using regular convex polyhedra, e.g. cube, octahedron, tetrahedron, icosahedron, dodecahedron. However, the number of projection angles is restricted to the number of vertices or faces of the polyhedra.



n₁
P₁
P₂
n₂
● intersection point between normal vector and surface polygon
○ vector starting point of the surface polygon

Fig. 10 Testing method for inwardly pointing normal vectors and outwardly pointing normal vectors.

3. Implementation and experimental results

We have conducted experiments on both computer-generated objects and a real object. The viewing directions are defined by the viewing sphere model, where the cameras are equally spaced on a sphere, looking inwards towards the world coordinates' origin. The images were taken at ten uniformly distributed viewing directions derived from an icosahedron.

3.1 Computer generated objects

A bull and hand bones were modeled in a computer. All the objects are opaque. With full prior knowledge of camera positions and parameters, a set of 10 images of each object was simulated using the viewing directions derived from an icosahedron.

3.1.1 Bull

The opaque bull model, its images and the corresponding apparent contours are shown in Fig. 11. Fig. 12(a) and 12 (b) display the reconstructed frontier points, secondary vertices and triangular facets respectively. It can be seen that, approximately, the frontier points tend to cluster closely around high curvature parts of the surface. The original bull model and the reconstructed surface of the bull are shown in Fig. 13. The error distance distribution, measured from the shortest distances from centers of the reconstructed triangular facets to the original surface, can be found in Fig. 14.

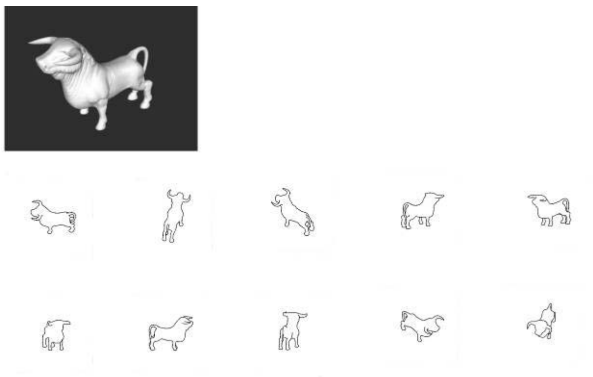
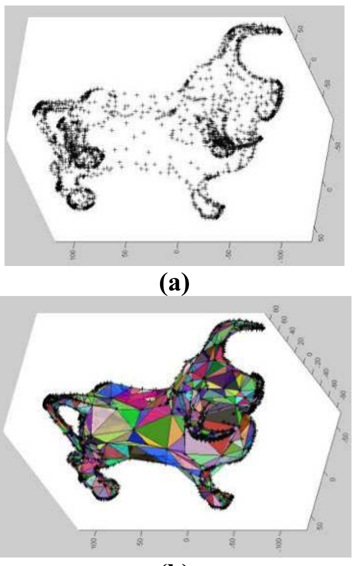


Fig. 11 Computer generated test object: an opaque bull model and its corresponding apparent contours derived from the 10 images taken from ten viewing directions.



(b)

Fig. 12 The bull model reconstructed from the 10 apparent contours in Fig. 11. (a) The reconstructed 3D frontier points and secondary vertices where the points have approximately self-optimized spacing on the surface. (b) The reconstructed triangular facets seen from different viewpoints (each facet is shown as a semitransparent facet).

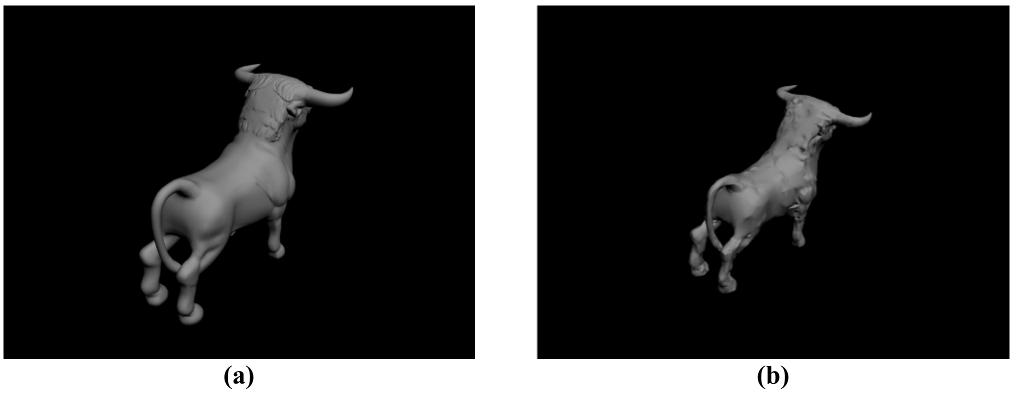


Fig. 13 (a) Original bull model (b) Reconstructed surface of the bull model.

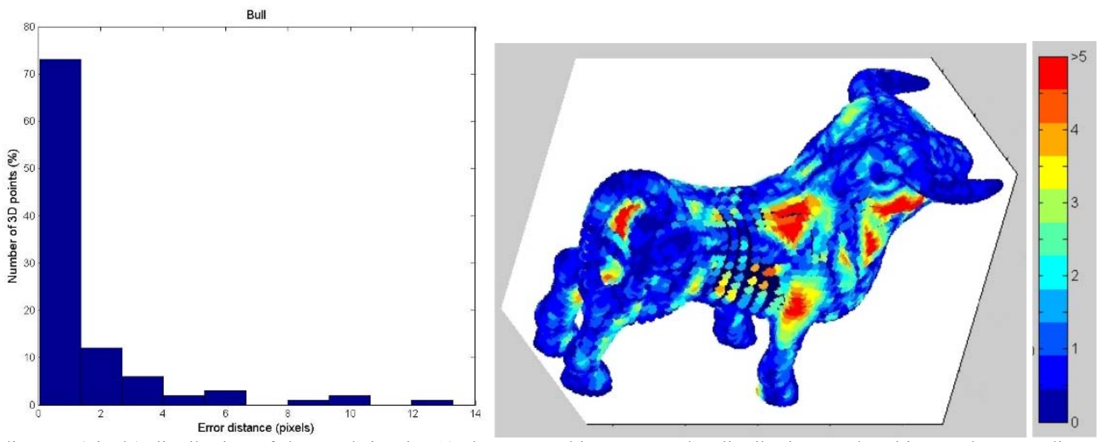


Fig. 14 Error distance (pixels) distribution of the result in Fig. 12 shown as a histogram and a distribution on the objects. The error distance distribution measured from the shortest distances from the centers of the reconstructed triangular facets to the surface of the original computer generated bull model.

3.1.2 Hand bones

The same method can also be applied to hand bone models. The computer-generated bones are shown in Fig. 15(a). The reconstructed 3D frontier points, secondary vertices and triangular facets of a bone are displayed in Fig. 15(b)–(c). The original model and the reconstructed surface are shown in Fig. 15(d)-(e) respectively. Two more bones are reconstructed and shown in Fig. 16(a)-(c). Fig. 17 shows all reconstructed bones. Note that each bone is reconstructed separately. The error distance distribution, measured from the shortest distances from centers of the reconstructed triangular facets to the original surface, can be found in Fig. 18.

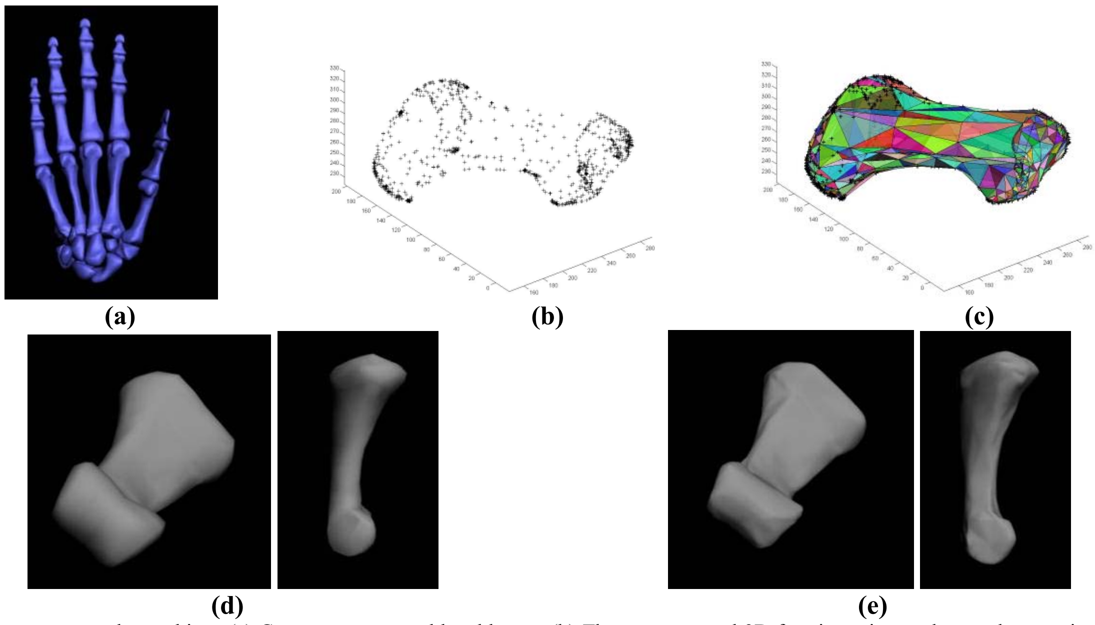


Fig. 15 Computer generated test object. (a) Computer generated hand bones. (b) The reconstructed 3D frontier points and secondary vertices of one of the bones, where the points have approximately self-optimized spacing on the surface. (c) The reconstructed triangular facets (each facet is shown as a semitransparent facet). (d) The original hand bone model seen from different viewpoints (e) The reconstructed surfaces of the bone model seen from different viewpoints.

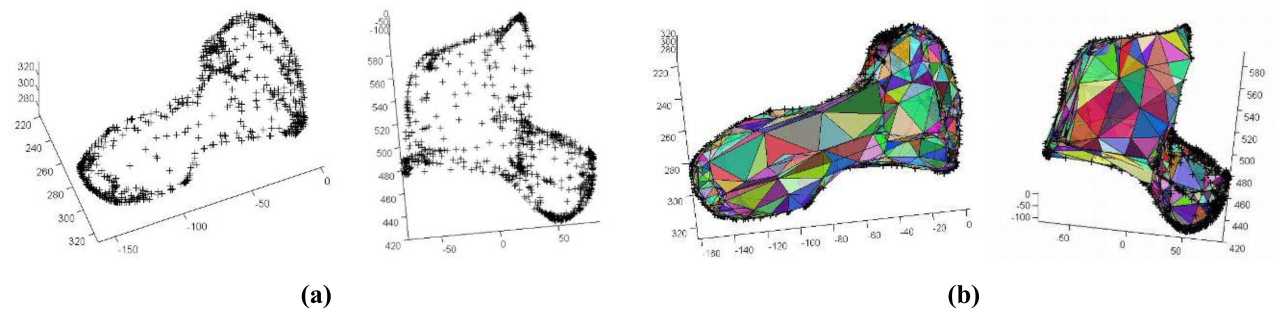
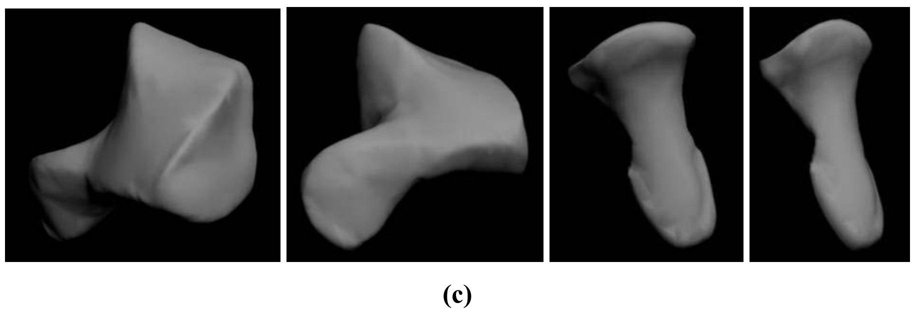
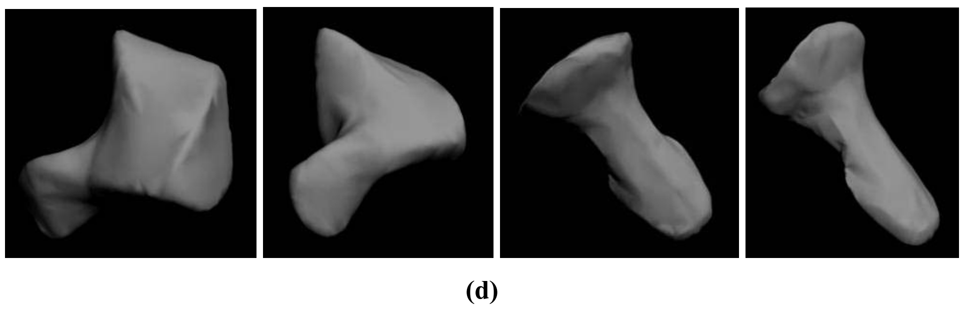




Fig. 16 Two more reconstructed bones. (a) The reconstructed 3D frontier points and secondary vertices. (b) The reconstructed triangular facets (each facet is shown as a semitransparent facet). (c) The original hand bone models seen from different viewpoints. (d) The reconstructed surfaces of the bone models seen from different viewpoints.

3.2 Real objects

In order to demonstrate that the proposed method is capable of being applied to real objects, we used physical objects, maracas, Fig. 19(a), and a rubber duck, Fig. 20(a). The maracas and the rubber duck were fixed on a clear plastic rod on a turntable. The required ten viewing directions were acquired by rotating the object and tilting the camera generating ten 2D images for the reconstruction process. The reconstructed 3D frontier points, secondary vertices and triangular facets are shown in Fig. 19(b)-(c) and Fig. 20(b)-(c). The rendered reconstructed surface is shown in Fig. 19(d) and Fig. 20(d). In this proof-of-concept experiment, the exact required viewing directions may not be achievable due to errors when rotating the object and tilting the camera. This may result in the some errors of the positions of the reconstructed 3D points. To reduce the viewing direction errors, multiple cameras could be fixed at desired viewing directions and all images are taken at the same time.

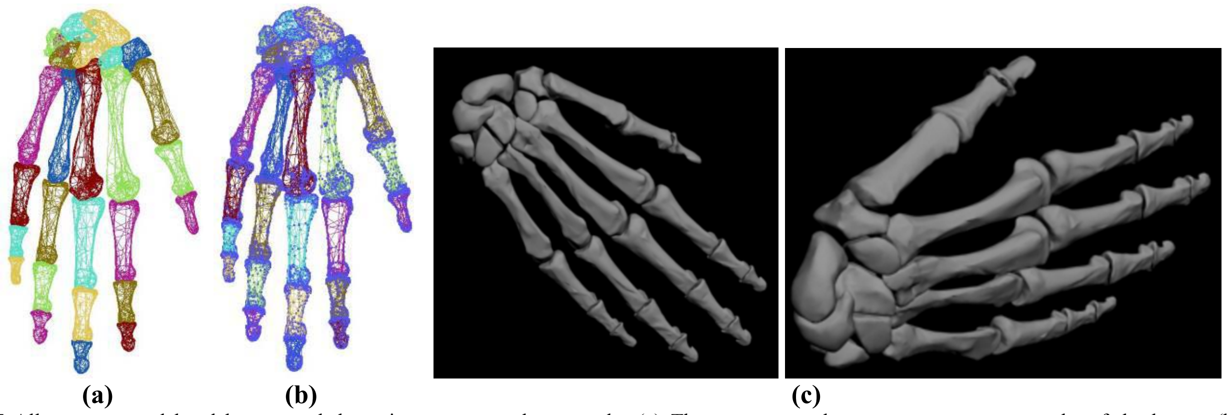


Fig. 17 All reconstructed hand bones, each bone is reconstructed separately. (a) The reconstructed contour generator networks of the bones. (b) The reconstructed 3D frontier points and secondary vertices superimposed on the contour generator networks. (c) The reconstructed surfaces of the bone models seen from different viewpoints.

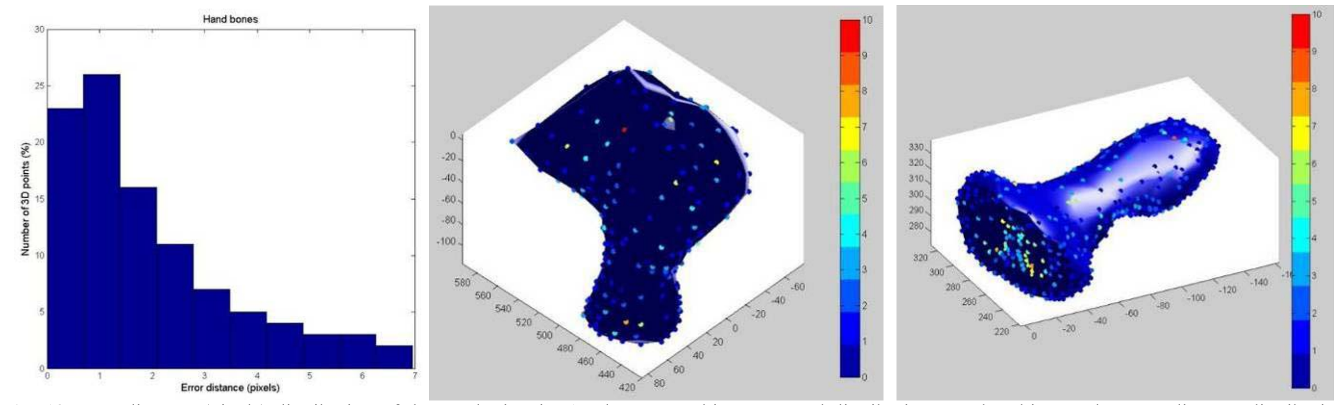


Fig. 18 Error distance (pixels) distribution of the results in Fig. 17 shown as a histogram and distributions on the objects. The error distance distribution measured from the shortest distances from the centers of the reconstructed triangular facets to the surface of the original computer generated bone models.

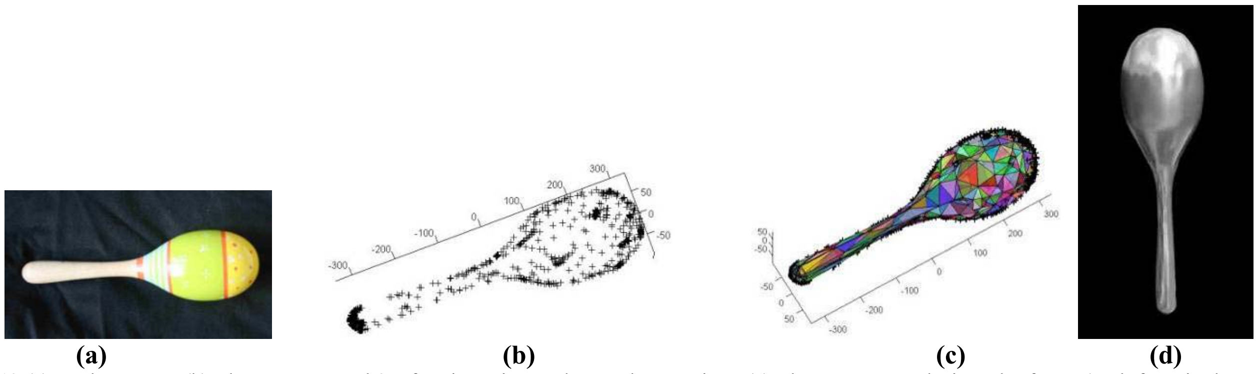


Fig. 19 (a) Real maracas. (b) The reconstructed 3D frontier points and secondary vertices. (c) The reconstructed triangular facets (each facet is shown as a semitransparent facet). (d) Rendered reconstructed surfaces.

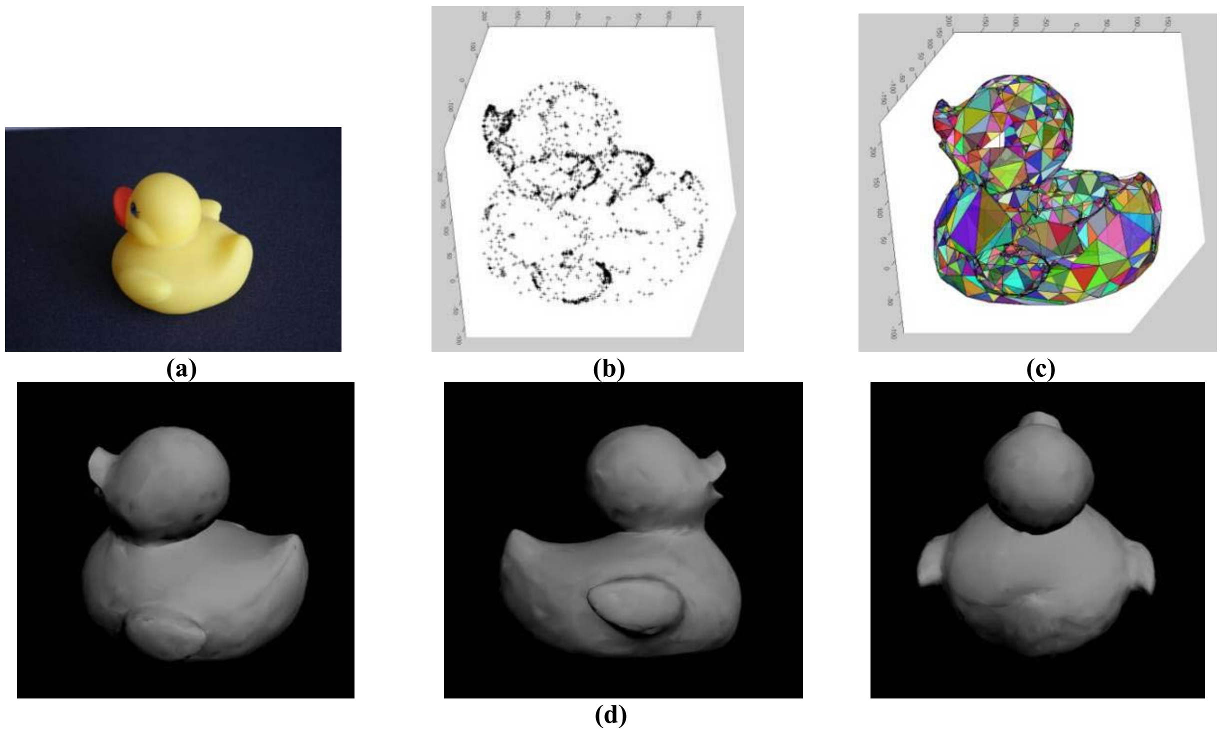


Fig. 20 (a) Real rubber duck. (b) The reconstructed 3D frontier points and secondary vertices. (c) The reconstructed triangular facets (each facet is shown as a semitransparent facet). (d) Rendered reconstructed surfaces.

4. Discussion and future work

In this paper we describe a method for reconstructing 3D surface points (frontier points and secondary vertices) and B-rep surface models (triangular facets) from a small number, e.g. 10, of 2D photographic images taken at equally distributed projection directions with full prior knowledge of camera configurations. The method has been tested with different types of objects and has shown that it is capable of reconstructing surfaces of 3D objects. Given the viewing directions are uniformly distributed; we qualitatively show that the frontier points tend to cluster closely on a highly curved part of a surface and widely spread on smooth or flat parts. The number and the distribution of frontier points depend only on the shape of the object, irrespective of its size. This creates a unique signature for each object which may also be used in 3D object recognition and measurement.

However, detecting correct and complete apparent contours is a very important factor in the whole reconstruction process. 2D images used in the proposed method must have sufficient quality so that the edge detection and fitting processes can extract and analytically represent all necessary apparent contours. There are still some problems found in the methods used in this work. Some apparent contours represented by spline curves are not smooth enough and some are twisted or folded at the ends creating wrong or redundant frontier points. Creating more perfect splines will be included in our future work. Surface's characteristics such as cusps, T-Junction and self-occlusions[44] can also be used in the apparent contour detection process. This will improve the quality of the resulting splines.

From the error distributions in Fig. 14 and Fig. 18, one could see that the majority of the triangular facets are close to the original surface. Nevertheless, there are some odd triangular facets which are displaced and some triangular facets are overlapped. This is due to some imperfections of the reconstructed contour generators. Currently, our surface reconstruction method cannot fully cope with those problems. We plan to investigate the problems more in detail and to produce a more robust method to eliminate those facets. A well-defined virtual elastic membrane or patch, which can be shrunk to wrap the reconstructed a contour generator network or a wireframe, will also be considered. This will guarantee that the final surface is always valid.

So far we have considered only single objects. Each object is reconstructed individually. It will be very challenging to automatically detect, identify and reconstruct multiple objects in a scene where objects are obstructed by other objects. This will be part of our future work.

At this stage, we concentrate only on empirical work to qualitatively demonstrate the proposed reconstruction method. Future work will include developing theoretical work on the distribution of frontier points and objects.

References

1. W. -S. Jang and Y.-S. Ho (2010) 3D Object reconstruction from multiple 2-D images, *3D Research* **1**(2): 02(2010)1-1 - 02(2010)1-5.
2. D. Cremers and K. Kolev (2011) Multiview stereo and silhouette consistency via convex functionals over convex domains, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **33**(6): 1161-1174.
3. K. Nurzynska, M. Kubo et al. (2011) Object Surface Reconstruction from Multi-View Images with Enhancement Driven by Image Forces, *SICE Annual Conference. Tokyo*.
4. K. Kolev, T. Brox et al. (2012) Fast Joint Estimation of Silhouettes and Dense 3D Geometry from Multiple Images, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **34**(3): 493-505.
5. Martin, W. N. and J. K. Aggarwal (1983) Volumetric descriptions of objects from multiple views, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **5**(2): 150-158.
6. C. Chien and J. Aggarwal (1986) Volume/Surface Octrees for the Representation of Three-Dimensional Objects, *Computer Vision, Graphics, and Image Processing* **36**(1): 100-113.
7. R. Szeliski (1993) Rapid octree construction from image sequences, *Computer Vision, Graphics and Image Processing CVGIP'93*.
8. E. Boyer (1996) Object models from contour sequences, *ECCV'96, Cambridge, U.K*.
9. W. Niem and J. Wingbermuhle (1997) Automatic reconstruction of 3D objects using a mobile monoscopic camera, *Int. Conf. Recent Advances in 3D Imaging and Modeling, Ottawa, Ont, Canada*.
10. A. W. Fitzgibbon, G. Cross et al. (1998) Automatic 3D model construction for turn-table sequences, *ECCV'98 Workshop on 3D Structure from Multiple Images of Large-Scale Environments, Freiburg, Germany*.
11. S. Sullivan and J. Ponce (1998) Automatic model construction, pose estimation, and object recognition from photographs using triangular splines, *Int. Conf. Computer Vision (ICCV), Bombay, India*.
12. K. Atsushi, H. Sueyasu et al. (2011) System for reconstruction of three-dimensional micro objects from mulitple photographic images, *Computer-Aided Design* **43**: 1045-1055.
13. Y. Iwashita, R. Kurazume et al. (2007) Patient-specific femoral shape estimation using a parametric model and two 2D fluoroscopic images, *ACCV'07 Workshop on multi-dimensional and multi-view image processing* 59-65.
14. R. Kurazume, K. Nakamura et al. (2007) 3D reconstruction of a femoral shape using a parametric model and two 2D fluoroscopic images, *IEEE International Conference on Robotics and Automation 2007* 3002-3008.
15. A. Szymczak, W. Hoff, et al. (2010) 3D shape from silhouettte points in registered 2D images using conjugate gradient method, *Medical Imaging 2010: Image Processing, SPIE*.
16. T. J. Cashman and A. W. Fitzgibbon (2012) What shape are dolphin? Buiding 3D morphable models from 2D images, *IEEE Transactions on Pattern Analysis and Machine Intelligence* DOI: http://doi.ieeecomputersociety.org/10.1109/TPAMI.2012.68
17. P. J. Giblin and R. Weiss (1987) Reconstruction of surfaces from profiles, *Proc. First Int'l Conf. Computer Vision, London*.
18. R. Cipolla and A. Blake (1992) Surface shape from the deformation of apparent contours, *International journal of computer vision* **9**(2): 83-112.
19. R. Vaillant and O. D. Faugeras (1992) Using extremal boundaries for 3D object modeling, *IEEE Transaction on pattern analysis and machine intelligence* **14**(2): 157-173.
20. C. S. Zhao and R. Mohr (1996) Global three-dimensional surface reconstruction from occluding contours, *Computer Vision and Image Understanding* **64**(1): 62-96.
21. J. -S. Franco and E. Boyer (2003) Exact Polyhedral Visual Hulls, *British Machine Vision Conference*.
22. K. Y. K. Wong and R. Cipolla (2004) Reconstruction of Sculpture From Its Profiles With Unknown Camera Positions, *IEEE Transactions on Image Processing* **13**(3): 381-389.
23. A. Sethi, D. Renaudie et al. (2004) Curve and Surface Duals and the Recognition of Curved 3D Objects from their Silhouette, *International journal of computer vision* **58**(1): 77-86.
24. T. Joshi, B. Vijayakumar et al. (1997) Hot curves for modelling and recognition of smooth curved 3D objects, *Image and Vision Computing* **15**: 479-498.

25. M. H. Kuo (2001) Automatic extraction of quadric surfaces from wire-frame models, *Computer and Graphics* **25**: 109-119.
26. K. Inoue, S. Kenji et al. (2003) Solid model reconstruction of wireframe CAD models based on topological embeddings of planar graphs, *Journal of Mechanical Design* **125**(3): 434-442.
27. J. -H. Gong, H. Zhang et al. (2007) Converting Hybrid wire-frames to B-rep models, *ACM Symposium on Solid and Physical Modeling, Beijing, China*.
28. S. Prakoonwit and R. Benjamin (2007) 3D surface point and wireframe reconstruction from multiview photographic images, *Image and Vision Computing* **25**(9): 1509-1518.
29. S. Prakoonwit and R. Benjamin (2009) Optimal 3D surface reconstruction from multiview photographic images, *IEEE International Conference on CyberWorlds. Bradford* 126-131.
30. R. Cipolla and P. Giblin (2000) Visual motion of curves and surfaces, *Cambridge, U.K., Cambridge University Press*.
31. P. R. S. Mendonca, K. -Y. K. Wong et al. (2001) Epipolar geometry from profiles under circular motion, *IEEE Transaction on pattern analysis and machine intelligence* **23**(6): 604-616.
32. R. Koren and Y. Yitzhaky (2006) Automatic selection of edge detector parameters based on apatial and statistical measures, *Computer Vision and Image Understanding* **102**(2): 204-213.
33. S. Kiranyaz, M. Ferreira et al. (2006) Automatic Object Extraction Over Multiscale Edge Field for Multimedia Retrieval, *IEEE Transactions on Image Processing* **15**(12): 3759-3772.
34. C. Tomasi and R. Manduchi (1998) Bilateral filtering for gray and color images, *6th International Conference on Computer Vision. Bombay, India*.
35. M. Lindenbaum and A. Berengolts (2000) A probabilistic interpretation of saliency network, *ECCV '00 Proceedings of the 6th European Conference on Computer Vision-Part II* 257-272.
36. M. Wertheimer (1938) Laws of organization in perceptual forms, *A Sourcebook of Gestalt Psychology. W. B. Ellis. New York, Harcourt Brace* 71-88.
37. T. -J. Cham and R. Cipolla (1999) Automated B-spline curve representation incorporating MDL and error-minimizing control point insertion strategies, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **21**(1): 49-53.
38. E. T. Y. Lee (1989) Choosing nodes in parametric curve interpolation, *Computer-Aided Design* **21**: 363-370.
39. H. Chen, K. -Y. K. Wong et al. (2006) Extracting surface representation from rim curves, *Computer Vision - ACCV 2006, Springer*.
40. A. Bjorck (1996) Numerical Methods for Least Squares Problems, *Linköping University, Linköping, Sweden*.
41. M. Mantyla (1986) Boolean operation of 2-manifolds through vertex neighborhood, *ACM Transactions on Computer Graphics* **5**(1): 1-29.
42. P. J. Giblin (1977) Graphs, surfaces and homology, *London, Chapman and Hall*.
43. M. Held (1998) Efficient and reliable triangulation of polygons. Computer Graphics International, *IEEE Computer Society* 633.
44. C. Liang, and K. -Y. K. Wong (2007) Robust recovery of shapes with unknown topology from the dual space, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **29**(12): 2205-2216.