# Free-form Object Reconstruction from Silhouettes, Occluding Edges and Texture Edges: A Unified and Robust Operator based on Duality

Shubao Liu, Kongbin Kang, Jean-Philippe Tarel, *Member, IEEE*, and David B. Cooper, *Fellow*

**Abstract**

In this paper, the *duality in differential form* is developed between a 3D primal surface and its dual manifold formed by the surface's tangent planes, i.e., each tangent plane of the primal surface is represented as a four-dimensional vector which constitutes a point on the dual manifold. The iterated dual theorem shows that each tangent plane of the dual manifold corresponds to a point on the original 3D surface, i.e., the "dual" of the "dual" goes back to the "primal". This theorem can be directly used to reconstruct 3D surface from image edges by estimating the dual manifold from these edges. In this paper we further develop the work in our original conference papers resulting in the robust differential dual operator. We argue that the operator makes good use of the information available in the image data, by using both points of intensity discontinuity and their edge directions; we provide a simple physical interpretation of what the abstract algorithm is actually estimating and why it makes sense in terms of estimation accuracy; our algorithm operates on all edges in the images, including silhouette edges, self occlusion edges, and texture edges, without distinguishing their types (thus resulting in improved accuracy and handling locally concave surface estimation if texture edges are present); the algorithm automatically handles various degeneracies; and the algorithm incorporates new methodologies for implementing the required operations such as appropriately relating edges in pairs of images, evaluating and using the algorithm's sensitivity to noise to determine the accuracy of an estimated 3D point. Experiments with both synthetic and real images demonstrate that the operator is accurate, robust to degeneracies and noise, and general for reconstructing free-form objects from occluding edges and

S. Liu, K. Kang and D.B. Cooper are with the Laboratory for Engineering Man/Machine System (LEMS), Division of Engineering, Brown University, Providence, RI 02912. Email: shubao_liu@brown.edu, {kk, cooper}@lems.brown.edu.

J.-P. Tarel is with DESE, Laboratoire Central des Ponts et Chaussées (LCPC), 58, Boulevard Lefebvre, 75015 Paris, France. E-mail: tarel@lcpc.fr.

texture edges detected in calibrated images or video sequences.

**Index Terms**

3D reconstruction robust to degeneracies and noise, duality in differential form, dual manifold, multi-view reconstruction, shape from silhouettes, shape from occlusions and textures, dynamic programming.

## I. Introduction

This paper, further developing the dual-space approach to 3D surface reconstruction from the edge curves in sets of calibrated images, includes the following five important contributions.

1) Estimation of a 3D point (expressed in homogeneous coordinates) on the primal surface is treated as determination of a 1-dimensional null space in a four dimensional space, where the other three dimensions are spanned by vectors that are dual to the surface to be reconstructed. The physical interpretation of this representation is that the estimated point lies on three planes in the primal space, each plane being one of the aforementioned dual vectors, and is therefore the intersection of three planes that have simple interpretations with respect to the 3D surface that is being estimated. The surface estimation is built around using both measured planes and measured rays that are tangent to the primal surface. Why is this a good approach to 3D reconstruction? The reason is that we assume the primal surface is piecewise $C^2$. A tangent plane contains surface normal and thus more information than does a ray. It can also be estimated more accurately in the presence of noise than can a ray. We also use interpolation among tangent planes and rays which incorporates additional information about assumed primal surface shape and additional smoothing for combating noise. In general, the type of interpolation employed can be tailored to the assumptions about the properties of the primal surface. This enables us to do accurate reconstruction of dense 3D points on a surface from sparse silhouettes and images of apparent contours of self occlusions.

2) The epipolar parametrization as incorporated into our surface reconstruction operator provides a fourth plane in our dual formulation which automatically handles degenerate cases such as frontier points.

3) The reconstruction algorithm operates on edges detected in the images. These edges can be silhouette edges, edges from self occlusion, or texture edges (due to albedo discontinuities, shadows, or lighting) on the 3D surface. The reconstruction algorithm treats these types uniformly,

i.e., it does not distinguish them by type.

4) Central to the surface reconstruction is establishing a two parameter parametrization on the dual manifold, the surface in the four dimensional space of vectors, each vector representing a plane tangent to the primal surface. Since the baseline between successive camera positions in a sequence may be large, a matching algorithm is introduced for creating this parametrization so that it is stable and effective. This problem is essentially that of finding pairs of edges, one in each of two successive images, that correspond.

5) A confidence measure is associated with each estimated 3D surface point. This provides a good mechanism for discarding estimated points which are likely to be outliers.

Our proposed operator is developed from shape from silhouettes, so we will first review the progress in this branch in detail. Due to the difficulty in extracting the occluding edges in an image, most of the previous works discuss silhouettes only — the occluding edges due to occlusion between object and background, although from the reconstruction algorithm viewpoint, all the occluding edges can be treated as the same. Recent progress in the field of computer graphics makes shape from occluding contours more interesting: in [1], Raskar et. al. designed a multi-flash camera to automatically extract depth edges from images for non-photorealistic rendering. Taubin et. al [2] immediately realized its importance and applied it to the field of 3D reconstruction. They used a refined method in [1] to automatically extract occluding contours, and with implicit surface fitting they showed that occluding contours can be used to recover complicated shape with high confidence. That study shows the rich geometric information contained in occluding edges.

Previous approaches to the problem of 3D reconstruction from silhouettes can be categorized into three groups: the discrete approach (also called visual hull approach), the differential approach, and the dual approach. The *discrete approach* takes the 3D reconstruction from silhouettes as a problem of computing the intersection of the volumetric cones back-projected from silhouettes in different views [3]–[10]. The main drawback of this approach is that it can not deal with the self-occluding edges and the reconstructed surface is not smooth. The *differential approach* is based on the assumption that the object surface is smooth enough to allow a differential analysis of the properties of the 2D silhouettes and of the 3D surface to be reconstructed [11]–[15]. In this method, strong assumptions are usually made about the motion of the camera and the local shape of the object to get a closed-form solution, and it cannot

deal with sparse views. The work in [2] mentioned above is based on this method with dense sampling. The main idea of *the dual approach* is to use the duality principle to have better insight in how to perform the 3D reconstruction by taking advantage of more information, especially the local information in tangent planes, and in the viewing ray. The duality property says that the primal surface and the dual manifold are equivalent in the sense that given one we can get the other through the dual mapping. The dual approach was first introduced with algebraic local curve estimation in [16]. Then the dual and algebraic approach was improved in [17] to handle singular cases.

Dual and differential approaches started to converge with the insights brought up in [18], [19], where the dual operator in its differential form was introduced for the first time for direct 3D reconstruction. Moreover, the arc-length parametrization of the silhouettes was used to rewrite this dual operator in a more robust way for 3D reconstruction from silhouettes. This study resulted in the dual operator with the ray constraint. One problem with the algorithm proposed in [18], [19] was that the search for the connectivity between views was relatively complicated. Thanks to [20], the epipolar constraint was introduced for improving the matching between views. However, only the epipolar constraint was used and the epipolar parametrization of the 3D object that comes from the differential approach was not fully used and fused with the dual approach. In this paper, we introduce a clear unification of the dual and differential approaches with new insights, which results in a robust differential dual operator.

None of those approaches address the problem of inferring objects' 3D structure from both *texture edges* and *occluding edges* in the same algorithm. Texture edges reflect the sharp changes in the intensity of the object surface, e.g., surface markings and texture patterns; Occluding edges are formed due to surface discontinuity in specific direction, i.e. occlusion, with silhouettes as a special case. The difference between these two kinds of edges include: (i) texture edges are viewpoint independent while the occluding edges are viewpoint dependent; (ii) texture edges usually provide detailed shape information, e.g., shape information about locally concave surfaces, while occluding edges usually can only provide rough but critical shape information and contain no information about the concave part of the surface; (iii) texture edge detections are usually sensitive to the environmental conditions, such as lighting, surface reflectance, shading, etc., while occluding edge detections are much more stable in images and video sequences, where the both image intensity discontinuity and motion discontinuity can be used to get the

occluding edges, e.g. [29]. In a word, these two kinds of edges complement each other to provide rich geometric information about the object surface. But until now in 3D reconstruction they are usually treated separately and with different methods. The texture contours on the surface, corresponding to texture edges in images, are usually recovered by triangulation and its variants, which are common techniques in stereo vision and multi-view vision [21]–[23]. In stereo vision, the existence of occluding edges creates significant difficulty in current procedures for matching two stereo images, and considerable effort has been put into developing matching algorithms robust to the disturbance of occluding edges, e.g. [24]–[28]. This makes the algorithms complex and computationally intensive.

Neither texture edges nor occluding edges can be used alone to handle all kinds of situations: for example, the concave part of an object surface cannot be recovered from contour edges; on the other hand, texture edges are not present on surfaces with uniform intensity or highly reflective mirror-like surfaces. The more general algorithm should incorporate both kinds of information to robustly and accurately infer the objects' 3D structure. There have been a few studies which try to fuse these two kinds of information, e.g. [30], [31]. In [30], a visual hull is constructed from silhouettes first, and then it is taken as the initial condition for the evolution of a PDE, which is evolved using the texture information. The core idea of [31] is fusing the texture and silhouettes information into a deformable model, where each kind of information corresponds to an external force, respectively texture driven force and silhouette driven force. In this paper, we fuse these two kinds of information using a different approach — handling them in one single reconstruction operator. With the developed robust differential dual operator, we build a connection between the differential dual approach developed in shape from silhouettes and the triangulation approach used in standard stereo reconstruction, or more generally, the multi-view reconstruction. This results in an operator capable of reconstructing 3D objects from general edge-maps, not only from silhouettes.

The remainder of this paper is organized as follows. In Sec. II, we discuss the duality in differential form and the robust differential dual operator. Then in Sec. III, we propose an algorithm for estimating each term in the operator from multiple images. And both the operator and the estimation algorithm are tested by comprehensive experiments in Sec. IV.

## II. Duality in Differential Form

The problem of 3D reconstruction from occluding edges taken in different views consists of making an inference about the shape of the 3D object so that it is as consistent with the given occluding edges as possible. When the camera is calibrated for the different views, it is easy from each occluding edge to estimate a series of planes tangent to the unknown 3D object by back-projecting the tangent lines of the edge curve. This fact leads us to study these sets of planes and introduce the duality as an important concept for doing the 3D reconstruction from silhouettes, which was first proposed in [16]. The types of duality we investigate include: duality between points and hyper-planes, duality between 2D curves, duality between surfaces, and how these duality properties fit well with the epipolar parametrization in the context of multiple view reconstruction. Let us start with the basic duality between points and hyper-planes.

Given a 3D point $\mathbf{P} \propto (x, y, z, 1)^t$ expressed in homogeneous coordinates, and a hyperplane $\mathbf{P}^* \propto (\mathbf{N}^t, d)^t$, where $\mathbf{N}$ is a unit vector perpendicular to the plane and $d$ is the distance from the plane to the origin, the point $\mathbf{P}$ is on the hyperplane if and only if $\mathbf{P}^{*t}\mathbf{P} = 0$. This last equation can be interpreted in two different ways:

- The first interpretation is to consider that this equation defines the set of points along the hyperplane $\mathbf{P}^*$.
- After transposition, we get $\mathbf{P}^t\mathbf{P}^* = 0$. This equation defines the set of hyper-planes going through $\mathbf{P}$. In the space of hyper-planes, the so-called dual space, $P^*$ is a point and $\mathbf{P}$ is a hyperplane. Thus in the dual space, the equation can also be interpreted in a very similar way as before in the primal space.

Instead of studying the simple duality between point and hyperplanes, to discuss the duality for curves and surfaces we need to express it in differential form. For this purpose we assume standard normalization, in homogeneous coordinates, for points, straight lines and planes. In the appendix, we prove that the choice of normalization method doesn't affect the reconstruction result in theory.

### A. Duality for 2D Curves

In 2D ($d = 2$), the duality between points and lines leads to the duality between a curve and the set of its tangent lines, the so called dual curve. Let us assume that the curve is parameterized

by $s$. The duality can be applied at every $s$ between a point $\mathbf{p}(s)$ along the curve and its tangent line $\mathbf{p}_T^*(s)$. We have

$$\mathbf{p}_T^{*\,t}(s)\mathbf{p}(s) = \mathbf{p}(s)^t\mathbf{p}_T^*(s) = 0. \tag{1}$$

However, $\frac{d}{ds}\mathbf{p}(s) = (\frac{dx}{ds}, \frac{dy}{ds}, 0)^t$, and by definition $\mathbf{p}_T^*(s) = (\mathbf{n}^t, d)^t$ with $\mathbf{n}$ perpendicular to the curve. Therefore,

$$\frac{d\mathbf{p}}{ds}(s)^t\mathbf{p}_T^*(s) = 0. \tag{2}$$

As a consequence of (1) and (2), vector $\mathbf{p}_T^*(s)$ is perpendicular to both $\mathbf{p}(s)$ and $\frac{d\mathbf{p}}{ds}(s)$. Therefore, the tangent line to the curve at $s$ is given by:

$$\mathbf{p}_T^*(s) = \alpha(s)\mathbf{p}(s) \times \frac{d\mathbf{p}}{ds}(s), \tag{3}$$

where $\times$ is the 3D vector cross product and the factor $\alpha(s)$ is introduced to account for the normalization. This equation can be written in another form using the null space notation. We denote $null(A)$ as the basis for the null space of matrix $A$, and $null(a, b, \cdots)$ as the basis of the null space of vectors $a, b, \cdots$. Then

$$\mathbf{p}_T^*(s) \propto null\left(\mathbf{p}(s), \frac{d\mathbf{p}}{ds}(s)\right), \tag{4}$$

where $\propto$ denotes equality up to a scale factor.

The set of tangent lines $\mathbf{p}_T^*(s)$ forms a curve parameterized by $s$ in the dual space. We call it the dual curve. It is known from the $19^{th}$ century that for a smooth enough primal curve (twice differentiable), from the dual curve only, which consists of all the tangent lines, it is possible to reconstruct the primal curve by applying the same operator that was applied to build the dual curve. More formally, at points where the surface is regular, we have in homogeneous coordinates:

$$\begin{aligned} \mathbf{p}(s) &= \alpha^*(s)\mathbf{p}_T^*(s) \times \tfrac{d\mathbf{p}_T^*}{ds}(s) \\ &\propto null\left(\mathbf{p}_T^*(s), \tfrac{d\mathbf{p}_T^*}{ds}(s)\right). \end{aligned} \tag{5}$$

Equations (4) and (5) tell us the duality role of the primal curve and the dual curve. *Different from the duality of point and hyperplane, they are the duality between one functional form (curve here, or surface later) and another functional form (dual curve here, or dual manifold later) expressed in the differential operator.* We call this kind of duality by the name "duality in differential form", or just duality for short. The proof of this duality principle for curves is detailed in the appendix.

In practice, to remove the unknown factor $\alpha(s)$ in the homogeneous representation, the point $\mathbf{p}(s)$ is always normalized such that its third component is one. And the line $\mathbf{p}_T^*(s)$ is normalized such that the vector of its two first components has norm one. In fact, surprisingly in the appendix, we prove in the theory that the choice of $\alpha(s)$ doesn't affect the reconstruction result.

### *B. Duality for 3D Surfaces*

In 3D ($d = 3$), the duality between points and planes leads to the duality between a surface $\mathcal{S}$ and its dual manifold $\mathcal{S}^*$, which is formed by the tangent planes of $\mathcal{S}$. Let us assume that the surface is parameterized by $s$ and $v$. The duality can be applied at every $(s, v)$ between a point $\mathbf{P}(s, v)$ along the surface and its tangent plane $\mathbf{P}_T^*(s, v)$. Similarly to the previous section, we have

$$\mathbf{P}(s, v)^t \mathbf{P}_T^*(s, v) = 0. \tag{6}$$

By definition, $\mathbf{P}_T^*(s, v)$ as a function of $(s, v)$, is tangent to the surface at $\mathbf{P}(s, v)$ and therefore we have

$$\frac{\partial \mathbf{P}}{\partial s}(s, v)^t \mathbf{P}_T^*(s, v) = 0, \tag{7}$$

and

$$\frac{\partial \mathbf{P}}{\partial v}(s, v)^t \mathbf{P}_T^*(s, v) = 0. \tag{8}$$

As a consequence of equations (6), (7), (8), vector $\mathbf{P}_T^*(s, v)$ is perpendicular to the three vectors $\mathbf{P}(s, v)$, $\frac{\partial \mathbf{P}}{\partial s}(s, v)$ and $\frac{\partial \mathbf{P}}{\partial v}(s, v)$.

Therefore, the tangent plane to the surface at $(s, v)$ is given by:

$$\begin{aligned} \mathbf{P}_T^*(s, v) &= \alpha(s, v)\overline{\mathbf{P}(s, v) \wedge \tfrac{\partial \mathbf{P}}{\partial s}(s, v) \wedge \tfrac{\partial \mathbf{P}}{\partial v}(s, v)} \\ &\propto null((\mathbf{P}(s, v), \tfrac{\partial \mathbf{P}}{\partial s}(s, v), \tfrac{\partial \mathbf{P}}{\partial v}(s, v)) \end{aligned} \tag{9}$$

where $\wedge$ and $^-$ are respectively the exterior product and complement operator defined in exterior algebra, also called Grassmann algebra [32]. The term $a \wedge b \wedge c$ can be interpreted as a span space of $a$, $b$ and $c$. Its complement $\overline{a \wedge b \wedge c}$ can be seen as a generalization of the cross product from 3D to 4D, which is also called the meet operation in [23] because of its geometric meaning. The set of tangent planes $\mathbf{P}_T^*(s, v)$ forms a dual manifold $\mathcal{S}^*$, also parameterized by $(s, v)$. Similarly to the previous section, for smooth enough primal surfaces the same operator can be used in

the dual manifold to derive the set of tangent planes and thus to reconstruct the primal surface. More formally, we have in homogeneous coordinates:

$$\begin{aligned} \mathbf{P}(s,v) &= \alpha^*(s,v)\overline{\mathbf{P}_T^*(s,v) \wedge \frac{\partial \mathbf{P}_T^*}{\partial s}(s,v) \wedge \frac{\partial \mathbf{P}_T^*}{\partial v}(s,v)} \\ &\propto null(\mathbf{P}_T^*(s,v), \frac{\partial \mathbf{P}_T^*}{\partial s}(s,v), \frac{\partial \mathbf{P}_T^*}{\partial v}(s,v)) \end{aligned} \tag{10}$$

The proof of the duality property for surfaces is detailed in the appendix.

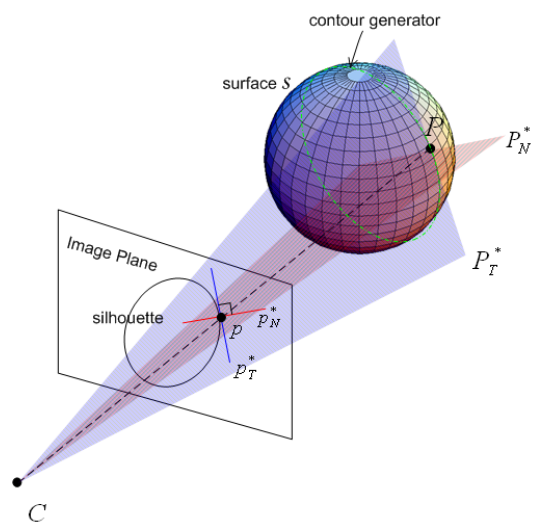



Fig. 1. The 3D point, $\mathbf{P}$, to be reconstructed, is constrained to be on the back-projection ray going through its image point $p$ and the camera center $\mathbf{C}$. The surface to be reconstructed is locally tangent to the plane $\mathbf{P}_T^*$, which is the plane going through $\mathbf{C}$ and the tangent line $\mathbf{p}_T^*$. $\mathbf{P}_N^*$ is the plane going through $\mathbf{C}$ and the normal line $\mathbf{p}_N^*$. Also note that usually $\mathbf{P}_N^*$ is not a normal plane of the surface $\mathcal{S}$, and so usually it is not perpendicular to $\mathbf{P}_T^*$.

Now let us see how to compute $\mathbf{P}_T^*(s,v)$ (expressed in the world coordinates, as are other 3D geometric entities) from the tangent line $\mathbf{p}_T^*$ (expressed in the pixel coordinates, as are other 2D geometric entities in the image plane) in the image and the projection matrix. For any point $\mathbf{P}$ on a surface $\mathcal{S}$, there is a tangent plane $\mathbf{P}_T^*$. It satisfies the following equations: (i) it goes through $\mathbf{P}$, i.e.

$$\mathbf{P}_T^{*\,t}\mathbf{P} = 0;$$

(ii) for any point $\mathbf{Q}$ on the tangent plane $\mathbf{P}_T^*$ (it satisfies $\mathbf{P}_T^{*\,t}\mathbf{Q} = 0$.), its image $\mathbf{q}$ (by projection matrix, we have $\mathbf{q} = M\mathbf{Q}$) lies on the tangent line, $\mathbf{p}_T^*$, of the occluding edge (i.e., $\mathbf{p}_T^{*\,t}\mathbf{q} = 0$). By substituting $\mathbf{q} = M\mathbf{Q}$ into $\mathbf{p}_T^{*\,t}\mathbf{q} = 0$, we have that

$$\mathbf{p}_T^{*\,t} M\mathbf{Q} = 0$$

holds for any $\mathbf{Q}$ lying on the tangent plane $\mathbf{P}_T^*$. Therefore, ${\mathbf{p}_T^*}^t M\mathbf{Q} = 0$ holds for any $\mathbf{Q}$ which satisfies ${\mathbf{P}_T^*}^t\mathbf{Q} = 0$. This implies that the two linear equations are equivalent. From this, we can derive that the coefficients of the linear equations, i.e., $\mathbf{P}_T^*$ and $M^t\mathbf{p}_T^*$, are equal up to a scalar multiplier $\beta$. That is,

$$\mathbf{P}_T^* = \beta M^t\mathbf{p}_T^* \propto M^t\mathbf{p}_T^*. \tag{11}$$

This relationship and the equation (10) give a clear way of reconstructing 3D points $\mathbf{P}$ from occluding edges. But it is not an optimal way, as shown in the next two subsections.

### *C. Ray Constraint*

Occluding edges in a image give us not only tangent planes but stronger information — the viewing rays. The reconstructed points should lie on these viewing rays. This is a constraint that is not utilized in (10), but can be used to improve accuracy generally and to handle an important kind of degeneracy in particular.

Assume that each occluding edge is parameterized by $s$ which is not necessarily the arc-length. Denote by $\mathbf{t}(s,v)$ the two component unit tangent vector of the silhouette at $s$, for viewing position $v$. We also introduce $\mathbf{n}(s,v)$ as the unit normal of the silhouette at $s$. We thus have by definition $\mathbf{t}^t(s,v)\mathbf{n}(s,v) = 0$ and $\mathbf{n}^t(s,v)\mathbf{n}(s,v) = 1$. By differentiating the last equation w.r.t. $s$, we deduce $\mathbf{n}^t(s,v)\frac{\partial\mathbf{n}}{\partial s}(s,v) = 0$. Since we are in 2D, $\frac{\partial\mathbf{n}}{\partial s}(s,v) = k(s,v)\mathbf{t}(s,v)$, where $k(s,v)$ is a scalar factor.

With the previous notation, the tangent line is:

$$\mathbf{p}_T^*(s,v) = \begin{bmatrix} \mathbf{n}(s,v) \\ -\mathbf{n}(s,v)^t\mathbf{p}(s,v) \end{bmatrix} \tag{12}$$

By differentiating w.r.t. $s$, we obtain:

$$\begin{aligned} \frac{\partial\mathbf{p}_T^*}{\partial s}(s,v) &= \begin{bmatrix} \frac{\partial\mathbf{n}}{\partial s}(s,v) \\ -\frac{\partial\mathbf{n}}{\partial s}(s,v)^t\mathbf{p}(s,v) - \mathbf{n}(s,v)^t\frac{\partial\mathbf{p}}{\partial s}(s,v) \end{bmatrix} \\ &= k(s,v)\begin{bmatrix} \mathbf{t}(s,v) \\ -\mathbf{t}(s,v)^t\mathbf{p}(s,v) + 0 \end{bmatrix}. \end{aligned}$$

Denote the vector in the previous equation as $\mathbf{p}_N^*(s,v)$, i.e.,

$$\mathbf{p}_N^*(s,v) = \begin{bmatrix} \mathbf{t}(s,v) \\ -t(s,v)^t\mathbf{p}(s,v) \end{bmatrix}. \tag{13}$$

From its definition, $\mathbf{p}_N^*(s,v)$ is the line perpendicular to the tangent line and crossing the curve at the same position. By taking the derivative of (11) w.r.t. $s$, we obtain

$$\frac{\partial \mathbf{P}_T^*}{\partial s}(s,v) = \beta(s,v)M(v)^t\frac{\partial \mathbf{p}_T^*}{\partial s}(s,v) + \frac{\partial \beta}{\partial s}M(v)^t\mathbf{p}_T^*(s,v). \tag{14}$$

And by using (13), we deduce that $\frac{\partial \mathbf{P}_T^*}{\partial s}(s,v)$ is a linear combination of $M(v)^t\mathbf{p}_N^*(s,v)$ and $\mathbf{P}_T^*(s,v)$. Define

$$\mathbf{P}_N^*(s,v) \propto M(v)^t\mathbf{p}_N^*(s,v).$$

With the same derivation as $\mathbf{P}_T^*(s,v)$ in Sec. II-B, we can see that $\mathbf{P}_N^*(s,v)$ is the plane going through both the camera center and the normal line $\mathbf{p}_N^*$. Putting the above definition into (14), we can get that $\frac{\partial \mathbf{P}_T^*}{\partial s}(s,v)$ is a linear combination of $\mathbf{P}_N^*(s,v)$ and $\mathbf{P}_T^*(s,v)$. This means that for 3D reconstruction from silhouettes, in the operator (10), we can substitute $\mathbf{P}_N^*(s,v)$ for $\frac{\partial \mathbf{P}_T^*}{\partial s}(s,v)$. The operator (10) is thus rewritten as:

$$\mathbf{P}(s,v) \propto null(\mathbf{P}_T^*(s,v), \mathbf{P}_N^*(s,v), \frac{\partial \mathbf{P}_T^*}{\partial v}(s,v)). \tag{15}$$

These geometric quantities, including $\mathbf{p}$, $\mathbf{p}_T^*$, $\mathbf{p}_N^*$, $\mathbf{P}$, $\mathbf{P}_T^*$, $\mathbf{P}_N^*$, and their relationships are shown in Fig. 1.

The operator (15) is better than (10) in practice since it substitutes $\mathbf{P}_N^*$, which is easily estimated from the silhouettes, for the partial derivative vector $\frac{\partial \mathbf{P}_T^*}{\partial s}$. Moreover, notice that from (3) and (11),

$$\frac{\partial \mathbf{P}_T^*}{\partial s} \propto M(v)^t\frac{\partial \mathbf{p}_T^*}{\partial s} \propto M(v)^t\frac{\partial}{\partial s}(\mathbf{p} \times \frac{\partial \mathbf{p}}{\partial s})$$

is a second order derivative, while $\mathbf{P}_N^*$ is only first order. In practice, a second order derivative is often too noisy to be used. By substitution of a first order derivative, we get a more robust operator. Another practical interesting property is that it provides a solution for degenerate cases, e.g., reconstructing cylinders where $\frac{\partial}{\partial s}\mathbf{p}_T^* = 0$ for the vertical line, see Fig. 4. In operator (15), planes $\mathbf{P}_T^*(s,v)$ and $\mathbf{P}_N^*(s,v)$ specify the ray going through image point $\mathbf{p}$ and the camera center, and the intersection of this ray and plane $\frac{\partial \mathbf{P}_T^*}{\partial v}(s,v)$ specify 3D point $\mathbf{P}$. Note, (15) is also interpreted as the specification of 3D point $\mathbf{P}$ as the intersection of three planes. Notice that in [16], the ray constraint was already introduced in the algebraic approach of the 3D reconstruction from silhouettes in the same spirit; in [18], [19], it was incorporated into the differential operator, but in an indirect way.

### D. Epipolar Parametrization

In the previous subsection, differential operator (10) is rewritten in the better form (15). However, another source of improvement is to take full advantage of the epipolar parametrization of the surface introduced in [12]. In [20], the epipolar constraint was introduced into the dual approach but only to improve the matching between two views. In [12], the "t-parameter curves are defined to lie instantaneously in the epipolar plane defined by the ray and direction of translation". Here we use $v$ instead of notation $t$. With the above definition and the fact that the tangent vector of the "v-parameter curve" should lie on the tangent plane, we get the conclusion that the tangent vector of the "v-parameter curve" goes in the direction of the viewing ray. That is, with the epipolar parametrization, $\frac{\partial \mathbf{P}}{\partial v}$ is on the ray going through point $\mathbf{P}$ and the camera center $\mathbf{C}$, see Fig. 2. Since the ray is defined by the intersection of two planes $\mathbf{P}_T^*$ and $\mathbf{P}_N^*$, therefore ${\mathbf{P}_T^*}^t \frac{\partial \mathbf{P}}{\partial v} = 0$ and ${\mathbf{P}_N^*}^t \frac{\partial \mathbf{P}}{\partial v} = 0$. From the last equation and the derivative of ${\mathbf{P}_N^*}^t(s,v)\mathbf{P}(s,v) = 0$ w.r.t. $v$, we therefore deduce $\frac{\partial {\mathbf{P}_N^*}^t}{\partial v}(s,v)\mathbf{P}(s,v) = 0$. The vector $\mathbf{P}$ is thus also perpendicular to $\frac{\partial \mathbf{P}_N^*}{\partial v}$. This implies that a new operator can be proposed:

$$\begin{aligned} & \mathbf{P}(s,v) \\ \propto \quad & null\left(\mathbf{P}_T^*(s,v), \mathbf{P}_N^*(s,v), \frac{\partial \mathbf{P}_T^*}{\partial v}(s,v), \frac{\partial \mathbf{P}_N^*}{\partial v}(s,v)\right) \end{aligned} \tag{16}$$

The intuitive geometric interpretation of operator (16) is that the four planes, $\mathbf{P}_T^*$, $\mathbf{P}_N^*$, $\frac{\partial \mathbf{P}_T^*}{\partial v}$, $\frac{\partial \mathbf{P}_N^*}{\partial v}$ all go through one point that we want to recover. This relationship is illustrated in Fig. 3. The operator (16) is more robust than (15) because it uses more information and it is robust against the noise. Especially, it is still valid for the so called frontier points [33], which occur when the tangent plane is identical to the epipolar plane. In that case, $\frac{\partial \mathbf{P}_T^*}{\partial v}(s,v)$ degenerates to be zero, see Fig. 4. For this reason, this operator is called the *robust differential dual operator*, which is used in the following 3D reconstruction algorithm. Note that each term in the operator is estimated based on assumptions about the object shape.

Another very interesting property of this new operator (16) is that the 3D point reconstruction by classical triangulation can be seen as the a special case of (16). Indeed, if the two derivatives

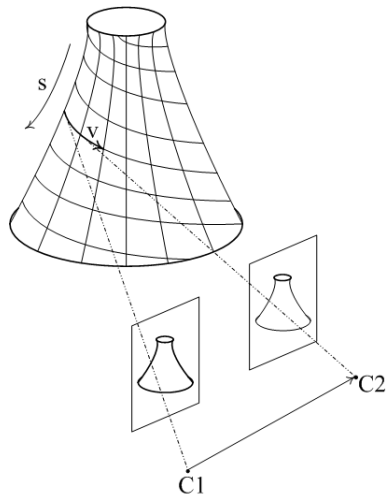



Fig. 2. Illustration of the epipolar parametrization of a 3D surface.

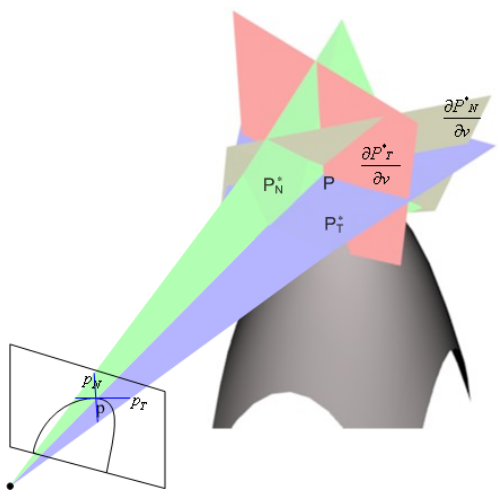
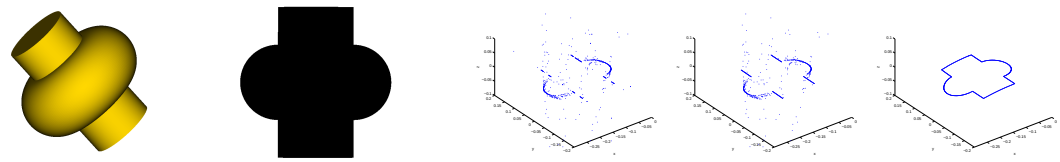



Fig. 3. The geometric interpretation of operator (16): four planes $\mathbf{P}_T^*$, $\mathbf{P}_N^*$, $\frac{\partial \mathbf{P}_T^*}{\partial v}$, $\frac{\partial \mathbf{P}_N^*}{\partial v}$ all go through one point $\mathbf{P}$.

in (16) are approximated by a simple first order difference scheme, we obtain:

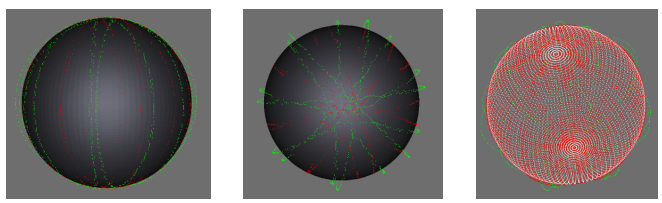
$$\begin{aligned} & \mathbf{P}(s,v) \\ \cong \quad & null(\mathbf{P}_T^*(s,v), \mathbf{P}_N^*(s,v), \frac{\mathbf{P}_T^*(s,v')-\mathbf{P}_T^*(s,v)}{v'-v}, \\ & \qquad \frac{\mathbf{P}_N^*(s,v')-\mathbf{P}_N^*(s,v)}{v'-v}) \\ \propto \quad & null\left(\mathbf{P}_T^*(s,v), \mathbf{P}_N^*(s,v), \mathbf{P}_T^*(s,v'), \mathbf{P}_N^*(s,v')\right) \end{aligned}$$

The last equation is equivalent of the classical triangulation by two rays characterized by $(\mathbf{P}_T^*(s,v), \mathbf{P}_N^*(s,v))$ and $(\mathbf{P}_T^*(s,v'), \mathbf{P}_N^*(s,v'))$, respectively. This relationship shows the con-



(a) (b) (c) (d) (e)

Fig. 4. In this figure, to remove the influence of the matching errors, we have worked with a synthetic object with surface of revolution. In this case, $\frac{\partial \mathbf{P}_T^*}{\partial v} = \frac{\partial M}{\partial v}\mathbf{p}_T^* + M\frac{\partial \mathbf{p}_T^*}{\partial v} = \frac{\partial M}{\partial v}\mathbf{p}_T^*$. Indeed, $\frac{\partial \mathbf{p}_T^*}{\partial v} = 0$ because of the same silhouettes in all view directions. Similarly, we have $\frac{\partial \mathbf{P}_N^*}{\partial v} = \frac{\partial M}{\partial v}\mathbf{p}_N^*$. (a) surface of revolution (b) the silhouette (c) the reconstruction of contour generator with operator (10); (d) the result using operator (15) which is better since it enforces the ray constraint, but it is still not good at frontier points; (e) the reconstruction using operator (16) is even more robust and nearly perfect.



(a) (b) (c)

Fig. 5. Comparison between results obtained with operator (16) and direct triangulation. Given 8 silhouettes of a sphere with radius $100cm$, the reconstructed contours with direct triangulation are shown in green points, and the reconstructed contours with the proposed new operator are shown in red. The average error for the proposed operator is $0.37cm$, while it is $4.14cm$ for the direct triangulation. From (a,b), visually we can see that the red points lie on the sphere, while the green points lie away from the sphere surface. (c) shows the interpolated point cloud (result from direct triangulation is not interpolated).

nection and difference between the proposed operator (16) and the direct triangulation. In some sense, (16) can be seen to be a generalization of classical triangulation to the case of occluding edges where continuity is enforced for the shape to be reconstructed. In practice, continuity assumptions are of main importance during interpolation of the 3D shape to be reconstructed. Fig. 5 shows the comparison between results obtained from (16) and direct classical triangulation. The proposed approach clearly achieves more accurate results.

More generally, suppose $\frac{\partial \mathbf{P}_T^*}{\partial v}$ and $\frac{\partial \mathbf{P}_N^*}{\partial v}$ are estimated as linear combinations of the samples of $\mathbf{P}_T^*(s, v_i)$ and $\mathbf{P}_N^*(s, v_i)$, $i = 0, 1, 2, \cdots$ and $v_0 = v$ as usually does. More specifically, suppose

$$\frac{\partial \mathbf{P}_T^*}{\partial v}(s, v) = \beta_0 \mathbf{P}_T^*(s, v_0) + \beta_1 \mathbf{P}_T^*(s, v_1) + \beta_2 \mathbf{P}_T^*(s, v_2) + \cdots ,$$

where $\{\beta_i\}$ are the set of scalar coefficients depending on the sample set $\{v_i\}$. Similarly for

$\frac{\partial \mathbf{P}_N^*}{\partial v}(s,v)$. Then if they are texture edges, that is, they all come from the same curve on the surface, the reconstruction operator is equivalent to

$$\begin{aligned}\mathbf{P}(s,v) \propto \quad & null(\mathbf{P}_T^*(s,v), \mathbf{P}_N^*(s,v), \mathbf{P}_T^*(s,v_1), \\ & \mathbf{P}_N^*(s,v_1), \mathbf{P}_T^*(s,v_2), \mathbf{P}_N^*(s,v_2), \cdots).\end{aligned}$$

This is the generalized triangulation for multi-view reconstruction.

The fact that triangulation is included in operator (16) has a very important consequence: it means that this operator should be able to reconstruct 3D surface from not only occluding edges but also the texture edges in the same setting. And different from the fusion approach in [30], [31], by using this operator, there is no need to discriminate these two kinds of edges which is often difficult. This will be shown in Sec. IV. This fact is very useful in 3D reconstruction from 2D images because:

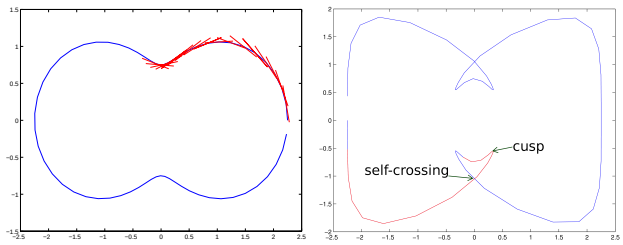
1) *In our approach the basic feature is the edge element (edgel) and its orientation instead of only edge point.* In the proposed operator based on this feature, the occlusion is no longer a trouble but of which an advantage can be taken.
2) There is no need to discriminate the edge types, and every type of edge is treated in exactly the same way.
3) By using the orientation information, although the reconstruction is also based on the local information it is much more stable than using just the point information. Indeed, the orientation contains very important local information.

### *E. The Structure of the Dual Manifold*

Actually the dual manifold crosses itself wherever the primal surface has bi-tangents as shown in Fig. 6. The self-crossing singular points on the dual manifold correspond to tangent planes in the primal surface that kiss the primal surface at more than one point. These kinds of singularities used to be a source of trouble for computing the derivatives because there are more than one possible tangent to these points. This problem is solved naturally using the connectivity information on the primal surface (the parametrization along $s$ and $v$). With this connectivity the derivatives can be computed at the neighborhood of self-crossing points. This is illustrated in a 2D case shown in Fig. 6.

There is another kind of singularity on the dual manifold: cusps. It happens when the primal curve has an inflexion point or when the primal surface has a parabolic point, and thus the dual

manifold is continuous but not differentiable at that point, as shown in Fig. 6 for the 2D curve case. Due to the nature of the proposed operator which is insensitive to the orientation along $v$, cusps are handled without difficulty.




(a) (b)

Fig. 6. Singularities: (a) primal curve with bi-tangents, (b) dual curve with self-crossing and cusps (The dual curve is shown as pedal curve, defined as $(\mathbf{n}_x d, \mathbf{n}_y d)$).

Until now, we have assumed that the primal surface is at least $\mathcal{C}^1$ continuous. What happens in the case where the primal surface is only $\mathcal{C}^0$ continuous, for example for a cube? At a surface ridge, under previous definitions there is no unique but rather many tangent planes. From occluding edges, we can get many tangent planes at these non-differentiable points. So we can still sample the dual manifold by including these planes such that the dual manifold stays continuous. Texture edges can be handled for exactly the same reason, as illustrated in Sec. IV.

## III. The Algorithm

From now on, we assume that the camera calibration is known, i.e, matrices $M(v)$ are given. In this section, we explain how to estimate each term in the operator (16) from multi-view observations (silhouettes, occluding-edges or general edge-maps) and how to perform the 3D reconstruction. Moreover, the operator's sensitivity to measurement errors is studied and this study results in a way to remove badly reconstructed points.

### A. Computing Derivative along $s$ Direction

The operator (16) involves two vectors which are not difficult to compute from a silhouette as explained in Sec. II-C:

$$\mathbf{P}_T^* = M(v)^t \mathbf{p}_T^*, \quad \mathbf{P}_N^* = M(v)^t \mathbf{p}_N^*.$$

To compute the tangent line $\mathbf{p}_T^*$ and tangent vectors $\mathbf{p}_N^*$, each edge curve is fitted by B-splines along the edgel linking. The required derivatives are then computed from the fitted curve.

### *B. Edge-map Matching*

The other two vectors in operator (16), the derivatives $\frac{\partial \mathbf{P}_T^*}{\partial v}$ and $\frac{\partial \mathbf{P}_N^*}{\partial v}$, should be carefully estimated. These two vectors are derivatives with respect to $v$. Different from the dense sampling in the $s$ direction, the sampling in the $v$ direction is usually sparse, and thus interpolation is required. The task is to interpolate the planes $(\mathbf{P}_T^*(s,v)$ and $\mathbf{P}_N^*(s,v))$ between views, then derivatives w.r.t. $v$ can be estimated. Contrary to the case in the $s$ direction, the connection between points along $v$ is not direct. A matching step between views is thus required.

In [20], given a point $\mathbf{p}(s,v)$ on the silhouette of the view $v$, the matching is performed by searching for the corresponding point $\mathbf{p}(s',v')$ in view $v'$ which is at the crossing of the epipolar line associated with $\mathbf{p}(s,v)$ and of the silhouette in the view $v'$. For a complicated object with branches, for instance, there is a matching ambiguity with this procedure: usually several points satisfy the previous constraints. This is why, following [8], it is better to compute the so-called viewing edge. This viewing edge can be defined as the segment of the ray $(\mathbf{P}_T^*(s,v), \mathbf{P}_N^*(s,v))$ going through $\mathbf{p}(s,v)$ and the camera center $\mathbf{C}(v)$ which is hidden by the object in all the other views.

The problem with the previous viewing edge approach is that we have to discriminate between texture edges and occluding edges which is a difficult task to perform in images. This problem drives us to perform matching between pairs of views using dynamic programming and then to remove badly matched points by testing reconstruction consistency during the 3D reconstruction, as described in the following sections. Silhouettes are closed curves, but in general the occluding edges and the texture edges are open curves. So it is not possible to use inside/outside discrimination for doing the matching as is the case for silhouettes.

In practice, we first rectify consecutive edge-map pairs to make the epipolar lines horizontal, then the matching is reduced to a one-dimensional alignment for each row. Each edgel in this row is described by three quantities, its relative position $l \in [0,1]$ within that row, its edge gradient magnitude $g \in [0,1]$, and edge orientation $r \in [0,\pi]$. Edgels in each row are represented as a vector. Then the matching in each row consists in aligning the vector pairs from two images. The basic idea is to introduce a similarity measure between two edgels and to find the best

alignment which makes the total similarities maximal. The optimization is performed using the Needleman-Wunsch algorithm [34], which guarantees optimal alignment. The similarity between two edgels is defined as:

$$s_{ij} = 1 - \sqrt{\gamma_l(\Delta l_{ij})^2 + \gamma_g(\Delta g_{ij})^2 + \gamma_r(\Delta r_{ij})^2},$$

where $\Delta l_{ij} = l_i - l_j$, $\Delta g_{ij} = g_i - g_j$, $\Delta r_{ij} = \frac{|r_i - r_j|}{\pi/2} - [\frac{|r_i - r_j|}{\pi/2}]$, $\gamma_l + \gamma_g + \gamma_r = 1$, and $\gamma_l, \gamma_g, \gamma_r$ are scalar parameters controlling the weighting of the edgel three properties, $[\cdot]$ is the round operation. In our experiments, we select $\gamma_l = 0.5, \gamma_g = 0.3, \gamma_r = 0.2$, and the skip penalty is chosen to be $d = 0.4$. More complex similarity measures may be needed for outdoor scenes.

### C. Computing Derivatives along $v$ Direction

After the matching between edge-maps in different views is performed, there are several ways for interpolating the rays $(\mathbf{P}_T^*(s,v), \mathbf{P}_N^*(s,v))$ between different values of $v$.

The simplest way is to directly interpolate tangent planes $\mathbf{P}_T^*(v)$ and normal planes $\mathbf{P}_N^*(v)$. For instance each plane can be interpolated using B-splines on each of its two angles and on the distance to the origin. The drawback is that the interpolation result depends on the origin location of the reference system.

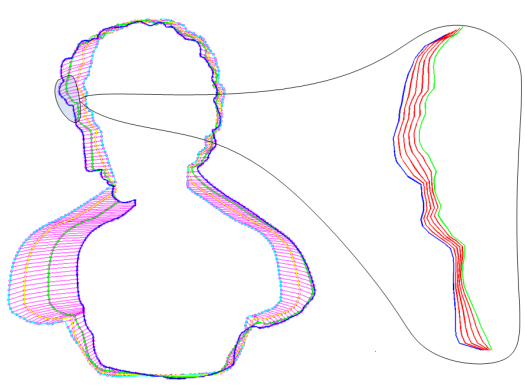


Fig. 7. Illustration of the edge-map interpolation. The matched results for four consecutive edge-maps are shown on the left. Parts of the interpolated edge curve through cubic splines are shown on the right marked with red color.

A different way of interpolation is to interpolate the camera motion and the change of edge-maps separately. First, the camera centers $\mathbf{C}(v)$ are interpolated in 3D and $m$ virtual camera centers are generated. Also from the interpolation, $\frac{\partial M}{\partial v}$ is computed. The edge curves

are interpolated along the $v$ directions with the matching results. Then $m$ virtual edge-maps are generated through equally spaced sampling the interpolated spline, see Fig. 7. With the interpolated curve (e.g., interpolated spline), $\frac{\partial \mathbf{p}_T^*}{\partial v}$ and $\frac{\partial \mathbf{p}_N^*}{\partial v}$ for each $s, v$ and also the new interpolated edgels can be estimated. Then $\frac{\partial \mathbf{P}_T^*}{\partial v}$ and $\frac{\partial \mathbf{P}_N^*}{\partial v}$ are computed as

$$\frac{\partial \mathbf{P}_T^*}{\partial v} \propto \frac{\partial M}{\partial v}\mathbf{p}_T^* + M\frac{\partial \mathbf{p}_T^*}{\partial v},$$
$$\frac{\partial \mathbf{P}_N^*}{\partial v} \propto \frac{\partial M}{\partial v}\mathbf{p}_N^* + M\frac{\partial \mathbf{p}_N^*}{\partial v}.$$

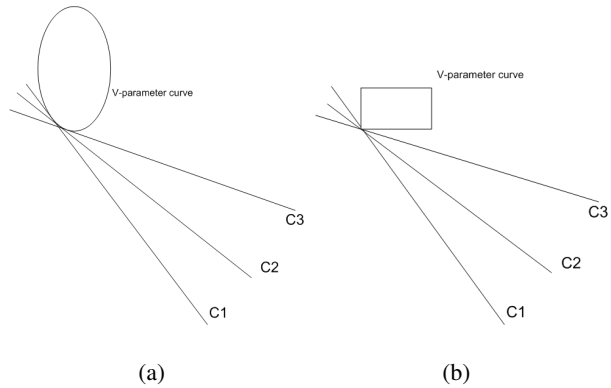
The interpolation requires knowing the $v$ values for each view. Different from the $s$ parameter, which is chosen to be the arc-length along the edge curve, the arc-length of the epipolar curve is not a good choice for the $v$ parameter. As an extreme case, for example, in the case of cube, the same edge is viewed in several images, where the arc-length along epipolar curve doesn't change but the tangent plane changes as discussed in Sec. II-E. In this case, the derivative of the tangent plane over arc-length along epipolar curve is not defined. The point is that since we are computing the change of the tangent planes which are points on the dual manifold, we should make $v$ parameter meaningful on the dual manifold, instead of on the primal surface. In the normal setup where the object depth is larger than the object size and the origin of the world coordinate locates around the object center, the change of the tangent plane is mainly influenced by the change of the angle between the camera center and the object center, and less influenced by the object shape as shown in Fig. 8. So usually we take the angle change between the object center (estimated approximately) and the camera center as the $v$ parameter.

### *D. Oriented Point Cloud Reconstruction*

For each edge point $\mathbf{p}(s, v)$, we have estimated the four vectors $\mathbf{P}_T^*$, $\mathbf{P}_N^*$, $\frac{d\mathbf{P}_T^*}{dv}$ and $\frac{d\mathbf{P}_N^*}{dv}$, which are involved in operator (16). In theory, the matrix composed of the above four vectors should be rank 3, with the null vector being the reconstructed point position. But in practice, due to errors from curve detection, interpolation and matching, the $4 \times 4$ matrix

$$\begin{aligned} & A(s,v) \\ = & \begin{bmatrix} P_T^*(s,v) & P_N^*(s,v) & \frac{dP_T^*}{dv}(s,v) & \frac{dP_N^*}{dv}(s,v) \end{bmatrix}^t \end{aligned} \tag{17}$$

is only of rank 3 approximately. In this case, the equation $A(s, v)\mathbf{P}(s, v) = 0$ can be robustly solved using the Singular Value Decomposition (SVD). In SVD, matrix $A$ can be decomposed




(a) (b)

Fig. 8. Illustration of the fact that tangent plane change is mainly influenced by the change of the angle between the camera center and the object center; the shape of the object has less influence on the tangent plane change. For simplicity, this figure shows only the 2D case.

as

$$A = U\Lambda V^t,$$

where $U^tU = I$, $V^tV = I$, and $\Lambda = diag(\lambda_1, \lambda_2, \lambda_3, \lambda_4)$ with $(\lambda_1, \lambda_2, \lambda_3, \lambda_4)$ being the singular values of the $A(s, v)$ matrix in the decreasing order. Also denote $\mathbf{u}_i$ as the $i$th column of $U$, and $\mathbf{v}_i$ as the $i$th column of $V$. Note that $\lambda_i \geq 0$, $i = 1, \cdots, 4$. Then the solution using SVD is to take the column of $V$ (called the right singular vector) corresponding to the smallest singular value in $\Lambda$, i.e., $\mathbf{v}_4$. And the result is optimal in the least square error sense [35].

The normal of the reconstructed points is useful information for surface reconstruction, where the orientation information can be utilized to fill holes and generate smooth surfaces. For the silhouette only, the orientation is easy to compute since the foreground and the background are known in images. But this is not applicable to self-occluding edges. We thus propose another approach based on the fact that if a point is on an occluding edge, then the corresponding viewing curve along $v$ is locally convex, as shown in Fig. 9. When the shape is locally convex, the rays on the previous and next views cross outside the object, and this point is used to orient the normal vector towards the outside.

### *E. Stability Analysis*

The matrix $A(s, v)$ is subject to the measurement errors of edge detection, inaccurate projection matrices, matching errors, errors of estimating the derivatives along $v$ direction. As a

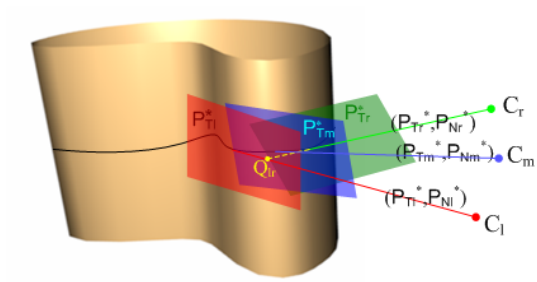



Fig. 9. Illustration of the normal orientation estimation using local convexity.

consequence, the study of the proposed operator's sensitivity is required. Assume that $A = \bar{A} + E$ is the noisy measurement of true $\bar{A}$. The question is how the right singular vector corresponding to the smallest singular value is affected when $\bar{A}$ changes to $A$?

Thanks to Wedin's theorem [36], this question can be answered with a simple upper perturbation bound. Here we follow [37], [38] to derive the upper bound of our problem. Denote the SVD of $\bar{A}$ as

$$\bar{A} = \bar{U}\bar{\Lambda}\bar{V},$$

where $\bar{U}^t\bar{U} = I$, $\bar{V}^t\bar{V} = I$, and $\bar{\Lambda} = diag(\bar{\lambda}_1, \bar{\lambda}_2, \bar{\lambda}_3, \bar{\lambda}_4)$ with $\bar{\lambda}_1, \bar{\lambda}_2, \bar{\lambda}_3, \bar{\lambda}_4$ in decreasing order. Because the rank of $\bar{A}$ is 3, $\bar{\lambda}_4 = 0$. Also denote $\bar{\mathbf{u}}_i$ as the $i$th column of $\bar{U}$, and $\bar{\mathbf{v}}_i$ as the $i$th column of $\bar{V}$.

Let $\Phi$ be the matrix of canonical angles (or principal angles) between two span spaces $\bar{U}_1 = (\bar{\mathbf{u}}_1, \bar{\mathbf{u}}_2, \bar{\mathbf{u}}_3)$ and $U_1 = (\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3)$. Let $\Theta$ be the matrix of canonical angles between $\bar{V}_1 = (\bar{\mathbf{v}}_1, \bar{\mathbf{v}}_2, \bar{\mathbf{v}}_3)$ and $V_1 = (\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3)$. Wedin's theorem is expressed in term of two following residuals:

$$R = \bar{A}V_1 - U_1\Lambda_1, \quad S = \bar{A}^T U_1 - V_1\Lambda_1$$

where $\Lambda_1 = diag(\lambda_1, \lambda_2, \lambda_3)$, $R$ and $S$ are $4 \times 3$ matrices. We have the following relationship between residuals and $E$:

$$\begin{aligned} ||R||_F &= ||(A - E)V_1 - U_1\Lambda_1||_F \\ &= ||EV_1 + 0||_F \\ &\leq ||E||_F, \end{aligned}$$

where $||\cdot||_F$ denotes the matrix's Frobenius norm. The same occurs for $S$: $||S||_F \leq ||E||_F$.

Due to $\bar{\lambda}_4 = 0$, Wedin's theorem states that

$$\begin{aligned}\sqrt{||sin\Phi||_F^2 + ||sin\Theta||_F^2} &\leq \frac{\sqrt{||R||_F^2+||S||_F^2}}{\min_{i=1,2,3}|\lambda_i - \bar{\lambda}_4|} \\ &= \frac{\sqrt{||R||_F^2+||S||_F^2}}{\lambda_3} \\ &\leq \sqrt{2}\frac{||E||_F}{\lambda_3}.\end{aligned}$$

The left-hand side of above equation is the combination of the canonical angles for the left and right singular subspaces; and the right-hand side is related to the perturbation. Generally speaking, the larger $\lambda_3$ is, the smaller the canonical angles $\Theta$ will be and the corresponding subspace will be more stable to noise.

With the fact that $\mathbf{v}_4$ and $\bar{\mathbf{v}}_4$ are respectively the complement subspace of $V_1$ and $\bar{V}_1$, we conclude that $\mathbf{v}_4$ is also stable against noise for large enough $\lambda_3$. As another interesting consequence of Wedin's theorem, we can get a simple bound on the angular perturbation $\phi$ between $\mathbf{v}_4$ and $\bar{\mathbf{v}}_4$:

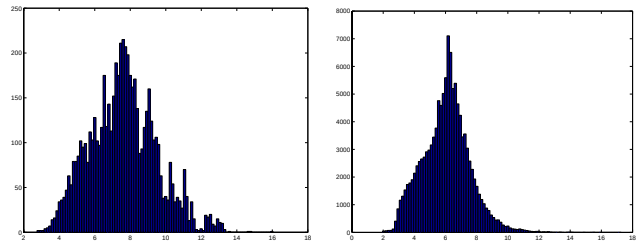
$$||sin\theta|| \leq \sqrt{2}\frac{||E||_F}{\lambda_3}. \tag{18}$$

This upper perturbation bound is of importance to remove outliers. This bound tells us that the operator (16)'s sensitivity to noise is directly related to the value of $\lambda_3$ as detailed in the next section.

### *F. Outliers Detection*

There are two main sources of reconstruction error, i.e., the estimation error of $P(s, v)$. One is the measurement noise, like the error of edgel position and the projection matrix; another is the singularity of the operator (16). From (17), we can see that matrix $A$ is directly related to the measurements. Then the noise in $A$, i.e. $E$, is a quantity directly related to the measurement noise. It is difficult to know the exact value of $||E||_F$ because the ground truth of $A$ is unknown, but $\lambda_4$ can be seen as an measurement of $||E||_F$, since we know the ground truth value of $\bar{\lambda}_4 = 0$. So it is reasonable to substitute $||E||_F$ as $\lambda_4$. For the singularity of the operator (16), the upper perturbation bound derived in Sec. III-E gives us the answer. When $\lambda_3$ is small, the reconstruction is sensitive to the measurement noise, while when $\lambda_3$ is large, the reconstruction is robust. From which we can see that the singularity of the reconstruction operator can be measured by $\lambda_3$. To perform outlier removal, we have to check both the consistency of the measurement and the reconstruction operator's singularity. That is, those two quantities should be combined to

get a criterion. Actually the combination is obtained by substituting $\lambda_4$ for $||E||_F$ in the upper bound, i.e., $\frac{\lambda_4}{\lambda_3}$. In practice, to set the threshold at the ratio $\frac{\lambda_4}{\lambda_3}$, we build the distribution of $-ln(\lambda_4) + ln(\lambda_3)$. The resulting histogram looks like a Gaussian probability density function (pdf) as shown in Fig. 10. Then the threshold can be set such that it removes $5\%$ (or some other specified percentage) of the samples on the left tail by setting the threshold to be $\mu - 1.65\sigma$, where $\mu, \sigma$ are respectively the mean and the standard deviation of the pdf.



(a) (b)

Fig. 10. (a) $-ln(\lambda_4) + ln(\lambda_3)$ of bust dataset (see Fig. 13); (b) $-ln(\lambda_4) + ln(\lambda_3)$ of bunny dataset (see Fig. 16)

After all the contour generators are recovered, we need to remove the outliers. Although the singular values of matrix $A$ can give us some information, it cannot remove all the outliers. Another way of detecting the outliers is through projecting the recovered point cloud to each image and checking their consistency with the silhouettes (this has been used in [2]). More specifically, if the image of some point lies outside any of the silhouettes, it is a outlier. These two tests of outliers complement each other and in our following experiments we use both of them to get clean point clouds.

### *G. Surface Reconstruction*

Given the point cloud, there are many well-developed algorithms to get a mesh or an implicit surface. In the following experiments, the surface is obtained by using the meshing algorithm developed in [39]. This algorithm is able to mesh unorganized 3D points on the surface of a regular or arbitrarily complex (free-form) 3D object, based on a shock scaffold representation which is derived from the medial axis representation. This medial scaffold resembles a full 3D Voronoi diagram when the input is point cloud. A series of *shock transformations* is applied

iteratively to the medial scaffold to simplify it, while on the other hand re-meshing the surface, in that the pruning of a medial branch implies filling the gap between surface points.

We are building surfaces to illustrate the good quality of the obtained 3D reconstructions despite that the chosen algorithm may not perfectly fit our particular kind of point clouds. In fact, the normal direction of each point is not used to generate smooth surfaces and to fill holes in the adopted algorithms. Presently, the orientation of each point is used only at the rendering stage to get a better visual result. Although there are other algorithms which fully utilize the normal information, they are not directly applicable to our case, since our point clouds are only partially oriented in the sense that occluding contours have normal direction but texture contours do not.

The final algorithm is summarized in Algorithm 1.

## IV. Experiments

In this part, with both real and synthetic data, comprehensive experiments have been performed to show various aspects of the proposed algorithm, including its ability to deal with both occluding edges and texture edges, to deal with degenerate cases, and its capability of recovering complex objects. To show the accuracy of the reconstruction, the distribution of the outlier points are shown through coloring each recovered point with its confidence value defined as a linear function of $conf = -ln(\lambda_4) + ln(\lambda_3)$, with red indicating outliers (when $conf = 0$) and blue indicating normal points (when $conf = 1$). To visualize the dual manifold in a 3D space, we exploit the fact that the points are normalized in the sense that the first three elements denote the plane's orientation vector $n_x, n_y, n_z$ and the fourth element denotes the negative of the distance between the tangent plane and the origin, $-d$. With the relation between orientation vector and its spherical coordinate $(\rho, \phi, \theta)$ [1], the dual manifold can be visualized in the spherical coordinate system. Here three real datasets, respectively David bust, hand and frog (thanks to [2].) and two synthetic datasets, Stanford bunny and textured horse (as shown in Fig. 11) are representative shapes of objects.

[1]The transformation relation is $d = \rho$, $n_x = sin(\phi)cos(\theta)$, $n_y = sin(\phi)cos(\theta)$, $n_z = cos(\phi)$

**Algorithm 1** 3D reconstruction from silhouettes, occluding edges and/or general edge maps

**Input**: image sequences $I_i, (i = 1, \cdots, N)$ and their calibration matrix $M_i, (i = 1, \cdots, N)$.

**Output**: 3D point cloud and surface.

**Procedure**:

1) Turn the images $I_i$ into edge maps $E_i$ with gradient and orientation.
   a) Get the edge sub-pixel position through edge detection, like Canny; Or get the silhouette curve through segmentation; Or other possible ways of converting images into useful edge-maps.
   b) Link the edge points and fit a spline for each edge curve.
   c) Get the orientation of each edge point through computing the derivative of the spline representation of edge curve (for instance with cubic *approximate* spline.)
2) Match the consecutive edge-maps.
   a) Rectify the consecutive edge-maps to make the epipolar lines horizontal in the rectified images.
   b) Match each row of the rectified edge-maps through one-dimensional alignment with dynamic programming, as discussed in Sec. III-B.
   c) Transform the matching results in the rectified edge-maps back to the original edge-maps.
3) Estimate each quantity in operator (16).
   a) Compute $\mathbf{p}_T^*$ and $\mathbf{p}_N^*$ with (12) and (13).
   b) Compute $\mathbf{P}_T^*$ and $\mathbf{P}_N^*$ with (11), and normalize them.
   c) Fit splines for $\mathbf{P}_T^*$ and $\mathbf{P}_N^*$ over $v$ direction, i.e., through the matched points, and compute their derivatives $\frac{\partial \mathbf{P}_T^*}{\partial v}$ and $\frac{\partial \mathbf{P}_N^*}{\partial v}$ as discussed in Sec. III-C (using for instance cubic *fitting* spline).
4) Reconstruct the 3D points with the operator (16) by using SVD as discussed in Sec. III-D. If the input are silhouettes or occluding edges, also compute the normal vector.
5) Detect and remove outlier points through re-projection to the image planes and confidence thresholding of the reconstructed points using singular values as described in Sec. III-F.
6) Generate surface from oriented point clouds as discussed in Sec. III-G or with other proper surface reconstruction algorithms.

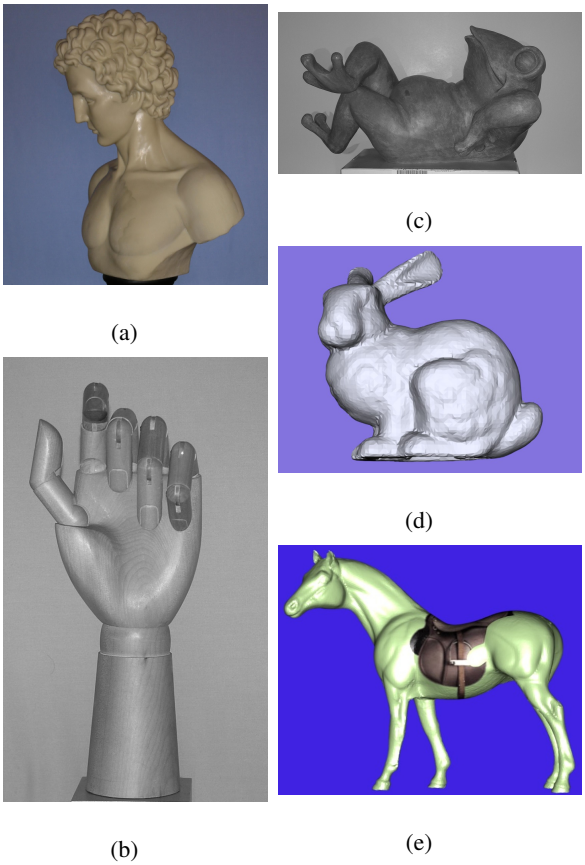



Fig. 11. Images of three real objects and 2 synthetic objects that we reconstruct from silhouettes or edges: (a) David bust (b) hand (c) frog (d) bunny (synthetic) (e) textured horse (synthetic).

### A. *Reconstruction from silhouettes*

As explained in Sec. II-E, our algorithm can deal with non-smooth objects, like a cube. Fig. 12 illustrates the different steps of the 3D reconstruction of a cube with 16 silhouettes.

Then more complicated objects are used to show the algorithm's ability to deal with free-form objects. In the next experiment, the David bust is put on a turn-table and 21 silhouettes are taken. The camera is fixed and calibrated with Zhang's method [40], and for each silhouette the pose of the object is known given the rotated angle. This is equivalent to fixing the object and to move the camera along a circular trajectory. Some examples of the silhouettes are shown in Fig. 13(a). Fig. 13 also displays the different stages of the 3D reconstruction, including the dual point cloud (b), the raw point cloud with color indicating the accuracy confidence (c), point cloud without interpolation (d), and point cloud after interpolation (e). As we can observe from Fig. 13(c),

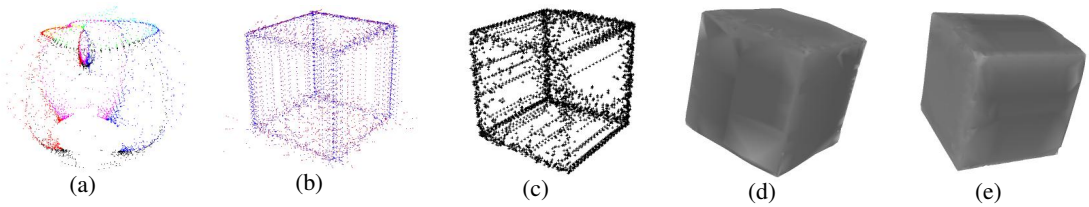



Fig. 12. Experiment on synthetic cube dataset with 16 silhouettes: (a) the dual point cloud computed from the silhouettes, (b) recovered raw point cloud with color indicating point confidence value (confidence increases from red to blue), (c) point cloud after removing the outliers, (d-e) recovered solid cube.

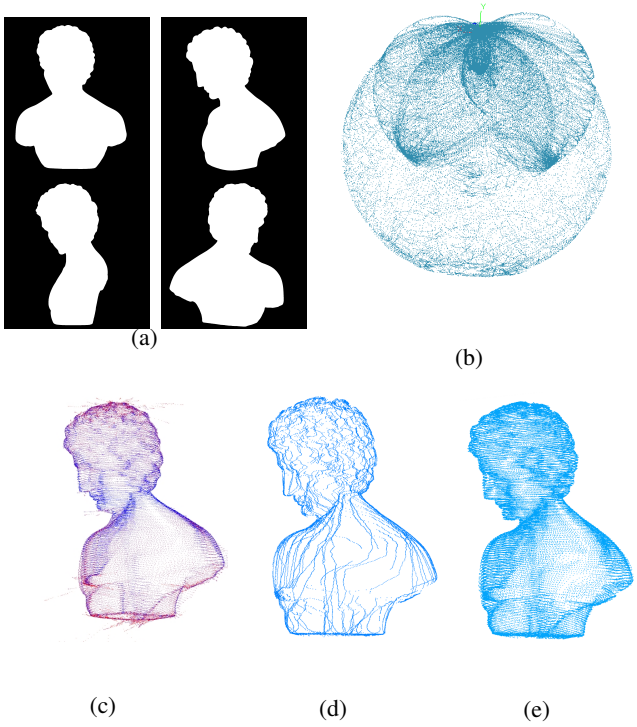
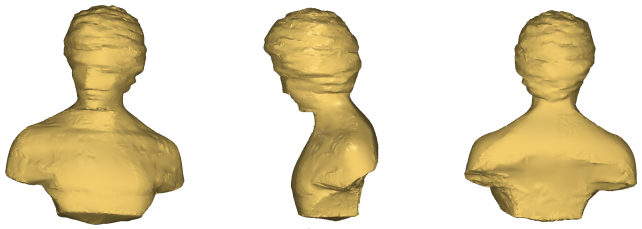



Fig. 13. Experiment on David bust dataset with 21 silhouettes: (a) sample silhouettes, (b) dual point cloud computed from silhouettes, (c) raw point cloud with color indicating point confidence value (confidence increases from red to blue), (d) recovered point cloud without interpolated points, (e) recovered point cloud with interpolation.

most of the outliers lie on the top, the bottom and also under the arm of the statue. This is due to the matching error where the degeneracy of the epipolar constraint occurs, and the sensitivity to matching errors due to the singular configurations between local shape at these points and the position of the camera as explained in Sec. III-E. Several snapshots of the reconstructed surface are shown in Fig. 14. As a comparison, the visual hull result generated by the EPVH algorithm [8] is shown in Fig. 15. The results are similar, since the object is nearly convex and



(a) (b) (c)

Fig. 14. Experiment on David bust dataset with 21 silhouettes: recovered solid object from different views.

(a) (b) (c)

Fig. 15. The result of EPVH approach with the same dataset as Fig. 14.

has nearly no self occluding parts.

Another experiment is simulated to take 20 silhouettes of the Stanford bunny with a virtual moving camera. As above, different stages of the reconstruction are shown in Fig. 16. Some snapshots of the reconstructed mesh are shown in Fig. 17. Observe that although the ear part of the bunny in different views is not correctly matched due to occlusion, the algorithm can still recovers it correctly, thanks to the outlier detection. The concave and nearly flat part of the bunny body are not recovered (see Fig. 16(d) and Fig. 16(e)) due to the inherent limitation of shape from silhouettes. This part is meshed with flat surface at surface reconstruction stage (see Fig. 17). The same data is used to generate the visual hull with EPVH algorithm as shown in Fig. 18. Comparing the results from two approaches, we can see that our proposed approach gives us generally more natural looking surfaces due to its interpolation capabilities, espcially for strong self-occluding parts.

### *B. Reconstruction from Occluding Edges*

With the reconstruction from occluding edges, we show the rich additional information provided compared with silhouettes only. First, the hand dataset is taken with a multi-flash to detect the occluding edges. Although in our system, there is no need to use this special device

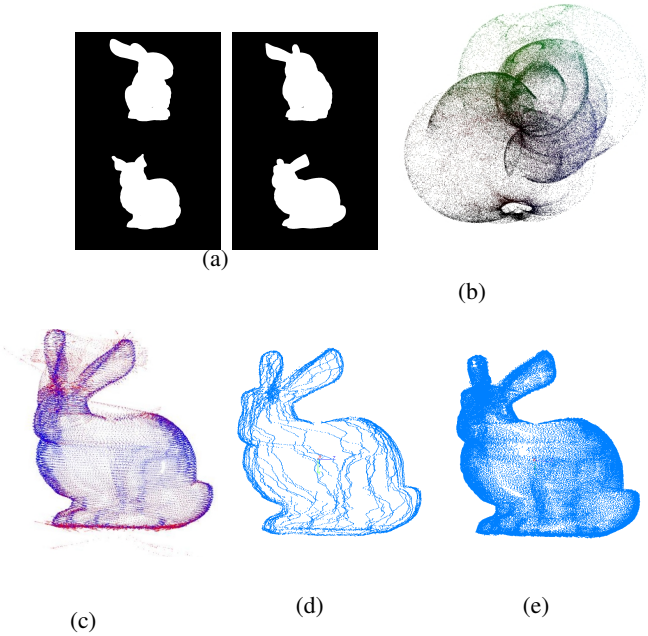



Fig. 16. Experiment on Stanford bunny dataset with 20 silhouettes: (a) sample silhouettes, (b) dual point cloud computed from silhouettes, (c) raw point cloud with color indicating point confidence value (confidence increases from red to blue), (d) recovered point cloud without interpolated points, (e) recovered point cloud with interpolation.

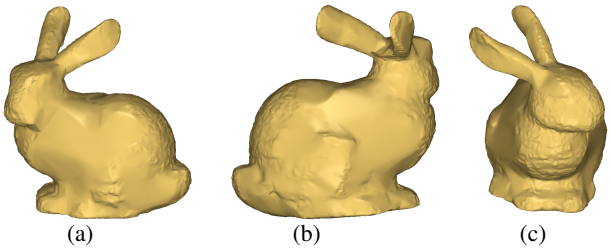
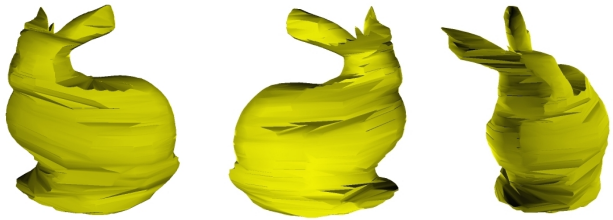



Fig. 17. Experiment on Bunny dataset with 20 silhouettes: recovered solid objects from different views.

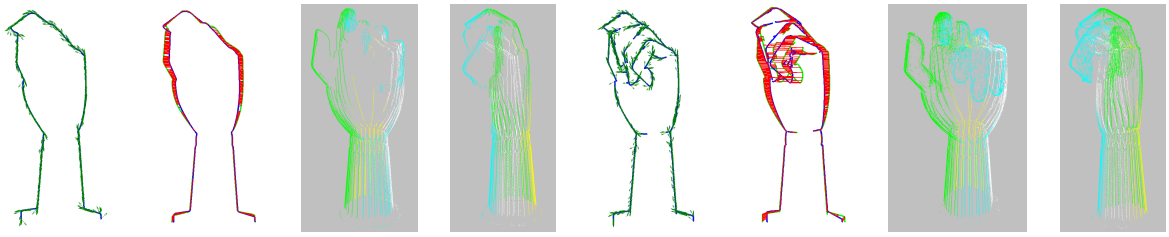
because our system deals with general edges. We use this dataset to observe the benefits of reconstructing from occluding edges instead of just silhouettes. The obtained point clouds are shown in Fig. 19. Figures (a-d) show results of reconstruction from silhouettes only, and figures (e-h) show results from occluding edges. Notice that with the occluding edges, our matching algorithm can still match the more complicated edge-map accurately and the matching results in much more complete reconstruction than with silhouettes only.

The effect of recovering shapes from all occluding edges instead of only silhouettes are better shown in Fig. 20, which shows the recovered surfaces from all occluding edges with our proposed



(a) (b) (c)

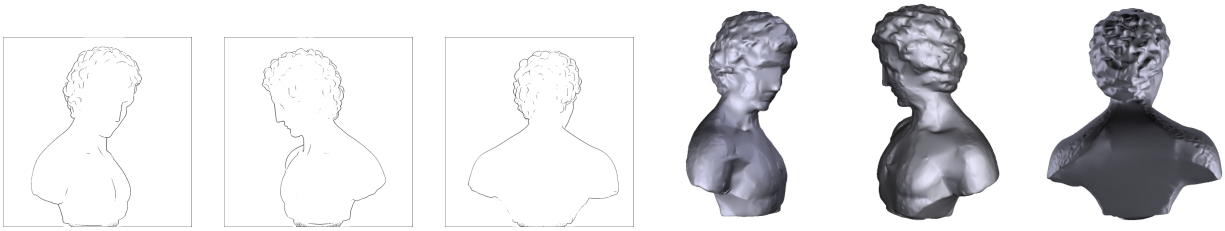
Fig. 18. Results of the EPVH approach with the same dataset as Fig. 17.



(a) (b) (c) (d) (e) (f) (g) (h)

Fig. 19. (a-d) are using silhouettes only, and (e-h) are using all occluding edges. (a) silhouette with tangent vectors, (b) two consecutive silhouette matched, (c,d) two views of the reconstruction from silhouettes, (e) occluding edges with tangent vectors, (f) two consecutive edge-maps matched (g,h) two views of the reconstruction using all occluding edges. 21 images used.

algorithm. From the results we can observe that the details of the object surface is recovered, like the complex hairs, the concave parts of the arm and etc., which are impossible from only silhouettes. The results for the frog dataset are shown in Fig. 21. Notice that the frog is an object with strong self-occlusion, and its geometry is more complex than the David bust and the Stanford bunny in terms of number of holes and strong concavities. The results show that the proposed algorithm can generate fairly good surfaces, based only on the edge information



(a) (b) (c) (d) (e) (f)

Fig. 20. (a-c) samples taken from 42 edge-maps; (d-f) recovered surface using the proposed algorithm.

for these types of objects, see Fig. 21. As a comparison, 27 silhouettes of the frog were used to generate the visual hull with EPVH. The results are not shown since they are not visually recognizable.

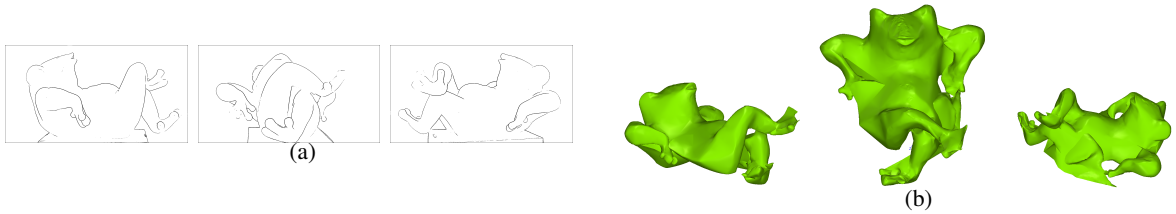



Fig. 21. (a) samples taken from 21 edge-maps; (b) different views of the reconstructed surface.

### C. Reconstruction from General Edge-Maps

As discussed in Sec. II-D, there is no need to discriminate between occluding edges and texture edges in the proposed algorithm. To illustrate this interesting property, we have generated synthetic views of a horse model textured with a saddle pattern on its body, see Fig. 11(e) for a sample. The reconstruction resulting from silhouettes only and edge-maps have been compared. Here we have used images taken from 7 different view-angles spanning half a circle around the object. Fig. 22 shows a matching between two views with silhouette only (b), and both occluding edges and texture edges (e). And the obtained reconstruction from silhouettes is shown in (c), as a comparison the obtained reconstruction from general edge-maps is shown in (f). From this figure, we can observe that with edges directly detected from images, we can obtain more information than from only silhouettes, especially the concave and flat part of the object (e.g., the horse body and the connecting part between the neck and the body). The algorithm we have proposed can easily deal with this without discriminating the edge types.

## V. Conclusions

Based on the duality for 3D surfaces, we have proposed a robust differential dual operator for performing 3D reconstruction from silhouettes, occluding edges and texture edges without distinguishing among them. This operator, based on measuring oriented edges rather than just image points of intensity discontinuity, makes full use of the “ray constraint” and the epipolar parametrization to ensure robustness to degeneracies and noise. The robust differential dual

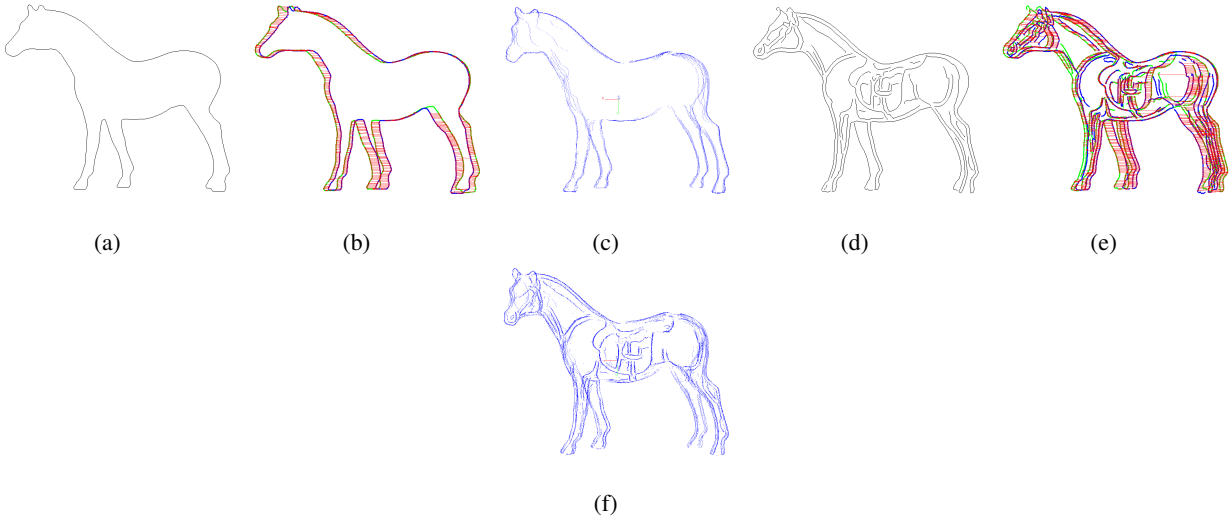



Fig. 22. (a) one of seven silhouette edge, (b) example of matching results, (c) reconstructed 3D point cloud from silhouettes only, (d) one of seven image edge-maps, (e) example of matching between edge-maps, (f) reconstructed 3D point cloud from edges detected from images

operator can be intuitively interpreted as intersection of four planes, all passing the point to be reconstructed. Two of the planes come directly from the image measurement, and the other two are estimated from interpolation of the first two planes along epipolar parametrization. Its desirable features are: easy to estimate, little sensitive to perturbations, able to tackle different kinds of singular cases of the 3D surface without modification, able to detect outliers and inaccurately reconstructed points. Last but not least, the operator is equivalent to the classical 3D triangulation when dealing with the texture edges. This operator is thus a very general operator that should be of fundamental use in computer vision. As a comparison, the visual hull approach uses only the information contained in the tangent plane, has no interpolation capability and cannot deal with self-occluding contours; On the other hand, the differential approach works only with rays, without using information contained in the tangent planes.

An algorithm is proposed for performing the 3D reconstruction from edge maps based on this new operator. Experimental results illustrate the high accuracy and good performance of the proposed approach for reconstruction of occluding contours on the 3D surface, as well as texture contours on the 3D surface.

There are a number of generalization of the proposed approach that would be of great value. One is to introduce some a priori surface shape information in order to better treat

the problem of large camera baseline between successive views. Another arises because the matching of consecutive edge-maps is treated as a one-dimensional alignment problem, which can be implemented with dynamic programming. But the limitation of this method is that the matching is only optimal locally in the sense that the matching of edge points in different rows are totally independent, without using their connection information. So a global matching of edge maps is a research topic for the future. A third is the surface reconstruction from point cloud when only partial orientation information is available.

## ACKNOWLEDGEMENT

This work was partial supported by NSF Grants NO. 0205477 and NO. BCS-9980091. The authors thank Professor Gabriel Taubin's group for providing their datasets, and their help for setting up the experimental device. The authors would also like to thank Ming-ching Chang for providing the mesh code and Dr. Matthew Brand for providing support for the initial version of the research.

## APPENDIX

### A. Proof of Duality for Curves

Following is the proof of the proposition that the dual of the dual manifold is the primal surface, see [18].

*Proof:* See equation (19).

■

### B. Proof of Duality for Surfaces

Following is the proof that the dual of the dual manifold is the primal surface. Here we omit the parameters to make the proof clear, e.g., $\mathbf{P}(s, v)$ is denoted by $\mathbf{P}$. In this proof, we use the Grassmann algebra and the duality properties between the exterior product (denoted as $\wedge$) and the regressive product (denoted as $\vee$). We denote by $^{-}$ the complement operator. Details about those operators can be found in [32], [41].

With the use of Grassmann algebra, we can easily generalize the proof from 2D curves to 3D surfaces. To do so, we first derive a rule which is similar to the 3D cross product rule used in

$$
\begin{aligned}
\mathbf{p}_T^*(s) \times \tfrac{d\mathbf{p}_T^*}{ds}(s) &= \left(\alpha(s)\mathbf{p}(s) \times \tfrac{d\mathbf{p}}{ds}(s)\right) \times \tfrac{d}{ds}\left(\alpha(s)\mathbf{p}(s) \times \tfrac{d\mathbf{p}}{ds}(s)\right) \\
&= \alpha(s)\left(\mathbf{p}(s) \times \tfrac{d\mathbf{p}}{ds}(s)\right) \times \left(\tfrac{d\alpha}{ds}(s)\mathbf{p}(s) \times \tfrac{d\mathbf{p}}{ds}(s) + \alpha(s)\tfrac{d}{ds}\left(\mathbf{p}(s) \times \tfrac{d\mathbf{p}}{ds}(s)\right)\right) \\
&= 0 + \alpha(s)\left(\mathbf{p}(s) \times \tfrac{d\mathbf{p}}{ds}(s)\right) \times \alpha(s)\tfrac{d}{ds}\left(\mathbf{p}(s) \times \tfrac{d\mathbf{p}}{ds}(s)\right) \\
&= \alpha^2(s)\left(\mathbf{p}(s) \times \tfrac{d\mathbf{p}}{ds}(s)\right) \times \left(\tfrac{d\mathbf{p}}{ds}(s) \times \tfrac{d\mathbf{p}}{ds}(s) + \mathbf{p}(s) \times \tfrac{d^2\mathbf{p}}{ds^2}(s)\right) \\
&= \alpha^2(s)\left(\mathbf{p}(s) \times \tfrac{d\mathbf{p}}{ds}(s)\right) \times \left(0 + \mathbf{p}(s) \times \tfrac{d^2\mathbf{p}}{ds^2}(s)\right) \\
&= \alpha^2(s)\det(\mathbf{p}(s), \tfrac{d\mathbf{p}}{ds}(s), \tfrac{d^2\mathbf{p}}{ds^2}(s))\mathbf{p}(s) \\
&= \alpha^*(s)\mathbf{p}(s)
\end{aligned}
\tag{19}
$$

using the 3d cross product rule $(a \times b) \times (a \times c) = det(a, b, c)a$.

the previous section, but which applies for 4D vectors. For 4D vectors $a$, $b$, $c$, $d$, we have:

$$
\begin{aligned}
& (a \wedge b \wedge c) \vee (a \wedge b \wedge d) \vee (a \wedge c \wedge e) \\
= \; & det(a, b, c, d)det(a, b, c, e)a.
\end{aligned}
$$

This rule is derived by applying two times the common factor axiom [32]:

$$(\alpha_m \wedge \gamma_p) \vee (\beta_k \wedge \gamma_p) = (\alpha_m \wedge \beta_k \wedge \gamma_p) \vee \gamma_p,$$

where $\alpha_m$, $\beta_k$, $\gamma_p$ are elements of grade $m$, $k$ and $p$ respectively, and with $m+k+p = 4$. Notice that in the following, we also use the following property for 4D vectors:

$$(a \wedge b \wedge c \wedge d) = det(a, b, c, d).$$

and the anti-commutativity axiom:

$$(a \wedge b) = -(b \wedge a).$$

*Proof:*

$$
\begin{aligned}
& (a \wedge b \wedge c) \vee (a \wedge b \wedge d) \vee (a \wedge c \wedge e) \\
= \; & (\underbrace{c}_{\alpha_1} \wedge \underbrace{a \wedge b}_{\gamma_2}) \vee (\underbrace{d}_{\beta_1} \wedge \underbrace{a \wedge b}_{\gamma_2}) \vee (a \wedge c \wedge e) \\
= \; & (c \wedge d \wedge a \wedge b) \vee (a \wedge b) \vee (a \wedge c \wedge e) \\
= \; & -det(c, d, a, b)(\underbrace{b}_{\alpha_1} \wedge \underbrace{a}_{\gamma_1}) \vee (\underbrace{c \wedge e}_{\beta_2} \wedge \underbrace{a}_{\gamma_1}) \\
= \; & -det(a, b, c, d)(b \wedge c \wedge e \wedge a) \vee a \\
= \; & det(a, b, c, d)det(a, b, c, e)a.
\end{aligned}
\tag{20}
$$

■

Similarly, we deduce a second rule for 4D vectors:

$$(a \wedge b \wedge c) \vee (a \wedge b \wedge d) \vee (a \wedge b \wedge e) = 0.$$

With all the previous properties, we are now able to prove that the dual of the dual manifold is the primal surface in 3D:

*Proof:* See equation (21). ■

$$\begin{aligned}
& \overline{\mathbf{P}_T^* \wedge \frac{\partial \mathbf{P}_T^*}{\partial s} \wedge \frac{\partial \mathbf{P}_T^*}{\partial v}} \\
= \; & \overline{\overline{\alpha\mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial \mathbf{P}}{\partial v}} \wedge \frac{\partial}{\partial s}\overline{\alpha\mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial \mathbf{P}}{\partial v}} \wedge \frac{\partial}{\partial v}\overline{\alpha\mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial \mathbf{P}}{\partial v}}} \\
= \; & -\left(\alpha\mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial \mathbf{P}}{\partial v}\right) \vee \frac{\partial}{\partial s}\left(\alpha\mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial \mathbf{P}}{\partial v}\right) \vee \frac{\partial}{\partial v}\left(\alpha\mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial \mathbf{P}}{\partial v}\right) \\
= \; & -\left(\alpha\mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial \mathbf{P}}{\partial v}\right) \vee \left(\frac{\partial \alpha}{\partial s}\left(\mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial \mathbf{P}}{\partial v}\right) + \alpha\frac{\partial}{\partial s}\left(\mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial \mathbf{P}}{\partial v}\right)\right) \\
& \vee \left(\frac{\partial \alpha}{\partial v}\left(\mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial \mathbf{P}}{\partial t}\right) + \alpha\frac{\partial}{\partial v}\left(\mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial \mathbf{P}}{\partial v}\right)\right) \\
= \; & 0 + 0 + 0 - \alpha^3\left(\mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial \mathbf{P}}{\partial v}\right) \vee \frac{\partial}{\partial s}\left(\mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial \mathbf{P}}{\partial v}\right) \vee \frac{\partial}{\partial v}\left(\mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial \mathbf{P}}{\partial v}\right) \\
= \; & -\alpha^3\left(\mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial \mathbf{P}}{\partial v}\right) \vee \left(0 + \mathbf{P} \wedge \frac{\partial^2 \mathbf{P}}{\partial s^2} \wedge \frac{\partial \mathbf{P}}{\partial v} + \mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial^2 \mathbf{P}}{\partial s \partial v}\right) \vee \\
& \left(0 + \mathbf{P} \wedge \frac{\partial^2 \mathbf{P}}{\partial s \partial v} \wedge \frac{\partial \mathbf{P}}{\partial v} + \mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial^2 \mathbf{P}}{\partial v^2}\right) \\
= \; & -\alpha^3\left(\mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial \mathbf{P}}{\partial v}\right) \vee \left(\mathbf{P} \wedge \frac{\partial^2 \mathbf{P}}{\partial s^2} \wedge \frac{\partial \mathbf{P}}{\partial v}\right) \vee \left(\mathbf{P} \wedge \frac{\partial^2 \mathbf{P}}{\partial s \partial v} \wedge \frac{\partial \mathbf{P}}{\partial v}\right) \\
& -\alpha^3\left(\mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial \mathbf{P}}{\partial v}\right) \vee \left(\mathbf{P} \wedge \frac{\partial^2 \mathbf{P}}{\partial s^2} \wedge \frac{\partial \mathbf{P}}{\partial v}\right) \vee \left(\mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial^2 \mathbf{P}}{\partial v^2}\right) \\
& -\alpha^3\left(\mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial \mathbf{P}}{\partial v}\right) \vee \left(\mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial^2 \mathbf{P}}{\partial s \partial v}\right) \wedge \left(\mathbf{P} \wedge \frac{\partial^2 \mathbf{P}}{\partial s \partial v} \wedge \frac{\partial \mathbf{P}}{\partial v}\right) \\
& -\alpha^3\left(\mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial \mathbf{P}}{\partial v}\right) \vee \left(\mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial^2 \mathbf{P}}{\partial s \partial v}\right) \vee \left(\mathbf{P} \wedge \frac{\partial \mathbf{P}}{\partial s} \wedge \frac{\partial^2 \mathbf{P}}{\partial v^2}\right) \\
= \; & 0 - \alpha^3 \det(\mathbf{P}, \tfrac{\partial \mathbf{P}}{\partial s}, \tfrac{\partial \mathbf{P}}{\partial v}, \tfrac{\partial^2 \mathbf{P}}{\partial s^2}) \det(\mathbf{P}, \tfrac{\partial \mathbf{P}}{\partial s}, \tfrac{\partial \mathbf{P}}{\partial v}, \tfrac{\partial^2 P}{\partial v^2})\mathbf{P} + \alpha^3 \det{}^2(\mathbf{P}, \tfrac{\partial P}{\partial s}, \tfrac{\partial \mathbf{P}}{\partial v}, \tfrac{\partial^2 \mathbf{P}}{\partial s \partial v})\mathbf{P} + 0 \\
= \; & \alpha^* \mathbf{P}.
\end{aligned} \tag{21}$$

## REFERENCES


[1] R. Raskar, K.-H. Tan, R. Feris, J. Yu, and M. Turk, "Nonphotorelistic camera: depth edge detection and stylized rendering using multi-flash imaging," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 679–688, 2004.

[2] D. Crispell, D. Lanman, P. G. Sibley, Y. Zhao, and G. Taubin, "Beyond silhouettes: Surface reconstruction using multi-flash photographing," in *Proc. 3DPVT'06*, 2006.

[3] A. Laurentini, "The visual hull concept for silhouette-based image understanding," *IEEE Trans. PAMI*, vol. 16, pp. 150–162, February 1994.

[4] E. Boyer and M.-O. Berger, "3d surface reconstruction using occluding contours," *Int. Journal of Computer Vision*, vol. 22, pp. 219–233, Mar 1997.

[5] S. Sullivan and J. Ponce, "Automatic model construction and pose estimation from photographs using triangular splines," *IEEE Trans. PAMI*, vol. 20, pp. 1091–1096, October 1998.

[6] K. N. Kutulakos and S. M. Seitz, "A theory of shape by space carving," *IEEE Trans. PAMI*, vol. 38, no. 3, pp. 199–218, 2000.

[7] S. Lazebnik, E. Boyer, and J. Ponce, "On computing exact visual hulls of solids bounded by smooth surfaces," in *Computer Vision and Pattern Recognition (CVPR '01)*, pp. 156–161, 2001.

[8] J.-S. Franco and E. Boyer, "Exact polyhedral visual hulls," in *Proceedings of the Fourteenth British Machine Vision Conference*, pp. 329–338, September 2003. Norwich, UK.

[9] J.-S. Franco, M. Lapierre, and E. Boyer, "Visual shapes of silhouette sets," in *Proc. 3DPVT'06*, 2006.

[10] L. Guan, S. Sinha, J.-S. Franco, and M. Pollefeys, "Visual hull construction in the presence of partial occlusion," in *Proc. 3DPVT'6*, June 2006.

[11] R. Cipolla and A. Blake, "Surface shape from deformation of apparent contours," *Int. Journal of Computer Vision*, vol. 9, no. 2, pp. 83–112, 1992.

[12] R. Cipolla and P. Giblin, *Visual Motion of Curves and Surfaces*. Cambridge University Press, 2000.

[13] P. J. Giblin and R. S. Weiss, "Reconstructions of surfaces from profiles," in *International Conference on Computer Vision (ICCV '87)*, pp. 136–144, 1987.

[14] R. Vaillant and O. Faugeras, "Using extremal boundaries for 3-d object modeling," *IEEE Trans. PAMI*, vol. 14, pp. 157–173, February 1992.

[15] K. Wong, P. Mendonca, and R. Cipolla, "Structure and motion estimation from apparent contours under circular motion," *Image and Vision Computing*, vol. 20, pp. 441–448, April 2002.

[16] K. Kang, J.-P. Tarel, R. Fishman, and D. B. Cooper, "A linear dual-space approach to 3d surface reconstruction from occluding contours using algebraic surface," in *International Conference on Computer Vision (ICCV '01)*, vol. 1, pp. 198–204, 2001.

[17] K. Kang, J.-P. Tarel, and D. B. Cooper, "A unified linear fitting approach for singular and non-singular 3d quadrics from occluding contours," in *HLK'2003*, (Nice, France), pp. 48–57, 2003.

[18] K. Kang, *Three-Dimensional Free Form Surface Reconstruction from Occluding Contours in a Sequence Images or Video*. PhD thesis, Division of Engineering at Brown University, Providence, RI, United States, May 2004.

[19] M. Brand, K. Kang, and D. B. Cooper, "Algebraic solution for the visual hull," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR '04)*, pp. 30–35, 2004.

[20] C. Liang and K.-Y. K. Wong, "Complex 3d shape recovery using a dual-space approach," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR '05)*, vol. 2, pp. 878–884, 2005.

[21] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, second ed., 2004.

[22] H. H. Baker and R. C. Bolles, "Generalizing epipolar-plane image analysis on the spatiotemporal surface," *Int. Journal of Computer Vision*, vol. 3, no. 1, pp. 33–49, 1989.

[23] Q.-T. Luong and O. Faugeras, *The Geometry of Multiple Images*. MIT Press, 2004.

[24] C. L. Zitnick and T. Kanade, "A cooperative algorithm for stereo matching and occlusion detection," *IEEE Trans. PAMI*, vol. 22, no. 7, pp. 675–684, 2000.

[25] Y. Nakamura, T. Matsuura, K. Satoh, and Y. Ohta, "Occlusion detectable stereo-occlusion patterns in camera matrix," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR '96)*, pp. 371–378, 1996.

[26] A. F. Bobick and S. S. Intille, "Large occlusion stereo," *International Journal of Computer Vision*, vol. 33, no. 3, pp. 181–200, 1999.

[27] S. B. Kang, R. Szeliski, and J. Chai, "Handling occlusions in dense multi-view stereo," in *IEEE Conference on Computer Vison and Pattern Recognition (CVPR '01)*, vol. 01, (Los Alamitos, CA, USA), p. 103, IEEE Computer Society, 2001.

[28] A. S. Ogale and Y. Aloimonos, "Shape and the stereo correspondence problem," *Int. Journal of Computer Vision*, vol. 65, October 2005.

[29] A. S. Ogale, C. Fermuller, and Y. Aloimonos, "Motion segmentation using occlusions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 6, pp. 988–992, 2005.

[30] R. Szeliski and R. Weiss, "Robust shape recovery from occluding contours using a linear smoother," *Int. Journal of Computer Vision*, vol. 28, no. 1, pp. 27–44, 1998.

[31] C. Esteban and F. Schmitt, "Silhouette and stereo fusion for 3d object modeling," *Computer Vision and Image Understanding*, vol. 96, pp. 367–392, December 2004.

[32] J. Browne, "Glassmann algebra - exploring applications of extended vector algebra with mathematica." Incomplete draft on http://www.ses.swin.edu.au/homes/browne/grassmannalgebra/book/, January 2001.

[33] P. Giblin and R. Weiss, "Epipolar fields on surfaces," in *Proc. 3rd European Conf. on Computer Vision*, vol. 1, pp. 14–23, May 1994.

[34] S. Needleman and C. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *J. Mol. Bio.*, vol. 48, pp. 443–453, 1970.

[35] G. H. Golub and C. F. VanLoan, *Matrix Computations*. Johns Hopkins University, 1996.

[36] P. Wedin, "Perturbation bounds in connection with singular value decomposition," *BIT*, vol. 12, pp. 99–111, 1972.

[37] G. W. Stewart, "Perturbation theory for the singular value decomposition," Tech. Rep. CS-TR-2539, 1990.

[38] F. M. Dopico, "A note on $\sin\theta$ theorems for singular subspace variations," *BIT*, vol. 40, pp. 395–403, 2000.

[39] F. Leymarie, *Three-Dimensional Shape Representation via Shock Flows*. PhD thesis, Division of Engineering at Brown University, Providence, RI, United States, May 2003.

[40] Z. Zhang, "Flexible camera calibration by viewing a plane from unknown orientations," in *International Conference on Computer Vision (ICCV '99)*, vol. 1, pp. 666–673, 1999.

[41] H. Barton, "A modern presentation of grassmann's tensor analysis," *American Journal of Mathematics*, vol. XLIX, pp. 598–614, 1927.