Interactive Visual Hull Refinement for Specular and Transparent Object Surface Reconstruction

Xinxin Zuo[†,‡], Chao Du[‡], Sen Wang[†,‡], Jiangbin Zheng[†] and Ruigang Yang[‡]
[†]Northwestern Polytechnical University, P.R.China [‡]University of Kentucky, USA
{xinxin.zuo,chao.du,sen.wang}@uky.edu, zhengjb@nwpu.edu.cn, ryang@cs.uky.edu

Abstract

In this paper we present a method of using standard multi-view images for 3D surface reconstruction of non-Lambertian objects. We extend the original visual hull concept to incorporate 3D cues presented by internal occluding contours*, i.e., occluding contours that are inside the object's silhouettes. We discovered that these internal contours, which are results of convex parts on an object's surface, can lead to a tighter fit than the original visual hull. We formulated a new visual hull refinement scheme* – Locally Convex Carving *that can completely reconstruct concavity caused by two or more intersecting convex surfaces.*

In addition we develop a novel approach for contour tracking given labeled contours in sparse key frames. It is designed specifically for highly specular or transparent objects, for which assumptions made in traditional contour detection/tracking methods, such as highest gradient and stationary texture edges, are no longer valid. It is formulated as an energy minimization function where several novel terms are developed to increase robustness.

Based on the two core algorithms, we have developed an interactive system for 3D modeling. We have validated our system, both quantitatively and qualitatively, with four datasets of different object materials. Results show that we are able to generate visually pleasing models for very challenging cases.

1. Introduction

3D reconstruction of specular or transparent objects is still a challenging problem in computer vision. Due to their non-Lambertian surface reflectance properties, establishing correspondences – a fundamental requirement for many 3D reconstruction algorithms – becomes difficult or even impossible. Therefore existing methods on reconstructing these difficult objects typically use additional constraints such as active illumination or known reference objects (e.g.,[12, 25, 8, 21, 31]).

In this paper we aim to reconstruct highly specular surfaces like glass sculptures and glossy trophies (such as these shown in Figure 1), from a multi-view images set casually captured with a hand-held camera, without using active illumination or reference objects (except a few markers for pose estimation). Naturally we decide to use a *visual hull*-based approach that does not require pixel correspondences. The fundamental limitation of the visual hull representation is that it is unable to model concavity. Through careful geometric analysis, we show that some type of concavity can actually be removed by using the *internal occluding contours*, i.e., occluding contours that are inside the object's silhouette. Based on that we present a new visual hull refinement method, which we refer to as *Locally Convex Carving* (LCC).

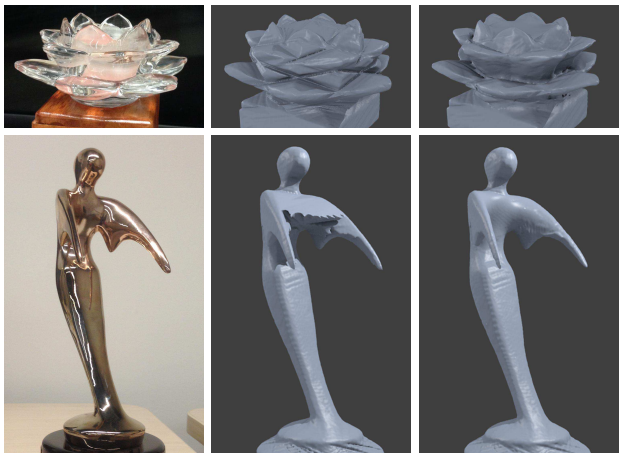


Figure 1: The reconstructed models for glossy trophy and glass sculpture. The left column shows the capture object; the middle column shows the reconstructed 3D model using visual hull; and the right column presents the reconstructed surface using our proposed method.

In addition we present a novel contour tracking algorithm designed specifically for highly specular/transparent objects. Different from most contour tracking algorithms that are based mainly on image gradient, we use the color statistics along contour to make the tracking more robust.

Based on these novel algorithms, we have developed a complete interactive system to achieve visually pleasing reconstruction results as shown in Figure 1. Our system takes in a number of calibrated images of the object to be modeled. The images are first segmented interactively using GrabCut [28]. A visual hull is reconstructed from the segmented images. Internal contours are labeled in a few key frames, then automatically prorogated to the remaining ones using our novel contour tracking algorithm. The results from tracking are sent to our unique LCC algorithm to refine the visual hull. Finally a simple depth propagation module can be used to interactively refine the remaining concave part.

2. Related Work

Our work is related to shape from silhouette (e.g., *Visual Hull*), contour tracking and reconstruction of non-Lambertian objects. We will briefly review related work in the following sections.

2.1. Occluding contours and visual hull

Occluding contours, or apparent contours have long been used for surface reconstruction [3, 7, 30]. They are the 2D projection of a set of surface points whose normals are orthogonal to the line of sight. The relationship between the 2D occluding contours and the 3D shape of the corresponding surface has been analyzed theoretically by Cipolla and Blake [7]. The very first attempts of using occluding contours for surface recovery can date back to 1990, where the reconstructed object tends to be simple in shape [3]. As the definition of visual hull is given in [18], researchers focus more on how to use *silhouettes*, which are the regions enclosed by occluding contours, to reconstruct 3D shapes from multi-view images. Various approaches have been developed for computing the visual hull of objects given silhouettes from multiple images [4, 5, 19, 9, 6]. Compared to typical stereo-vision, shape-from-silhouette does not require finding correspondences, which can be fragile in practice. However, visual hull cannot represent concavity. Therefore the silhouettes or visual hull constraints have been used in multi-view stereo reconstruction as complementary to photo-consistency cues, which can help in texture-less regions too [10, 13].

The occluding contours exploited in shape-from-silhouettes are actually *external* contours, which separate the object from the background. The use of *internal* contours has received less attention. Gargallo et el. [11] use internal contours while modeling visibility changes in mesh evolution framework. Results were only demonstrated on synthetic examples with highly distinctive colors. Recently Shan et al. combines all contours cues with a multi-view stereo reconstruction framework to get better result around object contours [29].

Our work is different from these reconstruction methods in that we focus on non-Lambertian objects and incorporate both external and internal contours to reconstruct a tighter shape than the visual hull.

2.2. Contour detection/tracking

Contour tracking is a traditional problem in image processing and computer vision area. Most approaches rely on active contour models [15] from which the initial contours will evolve to the desired object boundary. The traditional active contour framework involves parameterization of curves for performing visual tracking [27]. Other methods such as geometric active contours [26] for representing contours are the level set method where a contour is represented as the zero level set of a higher dimensional function. For general contour detection(e.g., [17]) or object boundary extraction, the cues in the energy function are based on image gradient or edge information combined with internal forces for smoothness. Contour tracking also relates to object tracking where objects are represented as contours to express the fine details of the object. For example, object motion and shape model prior [14] can be integrated in the formulation to better deal with noise and partial occlusion. Also, there are approaches that regard the tracking as a classification problem [2] and a classifier is trained to distinguish the object from the background. Then the contour pixels are extracted from the labeling [24].

While these methods typically track in 2D, the 3D cues conveyed by contours have be used in 3D object tracking as well [20]. Szeliski and Weisee [30] have exploited the 3D cues and used Kalman filtering to model the contours in images sequences so as to predict the contour in next frame. This method is able to differentiate stationary edges from occluding contours assuming diffuse objects.

It is beyond the scope of this paper to discuss the broad topic of tracking. We have not observed a previous method in contour detection and tracking that deals explicitly with highly specular objects. Many general contour tracking algorithms always assume that the contour is closed and the image gradient tends to be the most dominant external force to define the contour. However these assumptions are no longer valid in the cases this paper aims to address. With highly specular or transparent objects, texture edges are no longer stationary and gradient is no longer a strong indicator of contours due to specular highlight. Both factors make automatic and reliable detection/tracking of contours, in particular these inside silhouettes, very challenging.

2.3. Specular and transparent object reconstruction

Surface reconstruction of specular and transparent objects are challenging and methods using traditional stereo correspondence are not sufficient for these objects, since the complex reflection effects are not validate under the Lam-

bertian assumption. There are successful approaches [8, 21, 31] that use structured light methods relying on specialized patterns, where the surface depth or normal is computed by analyzing the captured patterns. In paper [8] a checkboard pattern is used and observed after distorted by the transparent objects; Liu et al. [21] design a set of frequency-based patterns. Zickler et al. [32] use Helmholtz stereopsis for surface reconstruction with arbitrary and unknown surface reflectance. The captured signal is transformed from the time domain into the frequency domain to solve the correspondence problem. As for specular objects, existing state-of-art methods can be broadly classified into two categories, namely *shape from specular flow* and *shape from specular correspondences* [22]. The first method assumes a known continuous motion and tries to track the dense specular flows, while the second one uses a reference plane with a known pattern as guidance to predict the unknown surface. The above methods all need careful setup and extra projector or light sensor. There are also methods that try to separate the specular reflection effects from diffuse reflection effects and use traditional photometric stereo method for surface reconstruction (e.g., [23]). These methods will have difficulty reconstructing highly specular or almost transparent objects like what we will present in this paper.

3. Our Approach

Our goal is to reconstruct highly non-Lambertian objects from a set of casually-captured multi-view images, without using any active lights. We assume that viewpoint for each image is known and the object has been segmented from the background. There are many existing techniques and tools that can achieve these two requirements. From the calibrated and segmented images, we first construct a visual hull of the 3D model using traditional volumetric visual hull reconstruction. The overall pipeline is shown in Figure 2, in which we develop a novel contour tracking method and a new visual hull refinement scheme that we referred to as *Locally Convex Carving*, finally concave areas are identified and interpolated using boundary conditions. In the next few sections we will present our methods in detail.

3.1. Terminology

An object's contour provides important clues about the object shape. Suppose a 3D object S is viewed by a camera. The object's silhouette image contains values that distinguish regions where the object is or is not present. Combined with calibration information for the camera, Each pixel in a silhouette defines a ray in scene space (denoted as E^3) that intersects the object S at some unknown depth. The union of these view rays for all pixels in the silhouette defines a generalized cone within which S must lie. If we are presented with multiple views of S, the intersection of these generalized cones from all views defines a volume in

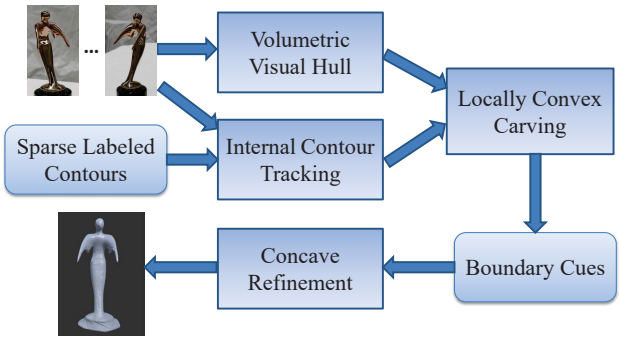


Figure 2: The system pipeline of our approach

E^3 that must contain S. As the number of the reference views, taking from different locations, goes to infinity, the intersection volume converges to the shape known as the objects *visual hull*, a term defined by Laurentini [18]. The visual hull, denoted as $VH(S)$, is guaranteed to contain the object S. In 2D, the visual hull is equal to the convex hull of the object (denoted as $CH(S)$). For 3D scenes, the visual hull is a tighter fit than the convex hull.

A less-known term, also defined by Laurentini [18], is the *internal visual hull* ($(IVH(S))$. $CH(S)$ segments E^3 into two regions. When all the reference views are taken outside $CH(S)$, $VH(S)$ is formed. If views are taken outside S, e.g., including concave areas in S but still outside S, $IVH(S)$ is formed. It can be shown that $IVH(S)$ is an even tigher fit than $VH(S)$, i.e., $S \leq IVH(S) \leq VH(S) \leq CH(S)$. However it is often difficult, if not impossible, to take pictures in concave areas in S. So $IVH(S)$ mostly remains a theoretical concept.

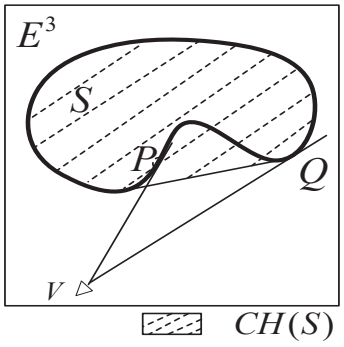


Figure 3: Illustration for occluding contours and visual hull. S is the real surface and $CH(S)$ is the convex hull which is same as visual hull ($VH(S)$) in 2d. P is the internal occluding contour point while Q is the external occluding contour point.

To explain our method, a few more terms need to be defined. As shown in Figure 3, a contour is a view-dependent concept, a point in a contour is a point on S for which a tangent line is intersecting S. The intersection point divide the tangent line into two segments. When both segments are outside $CH(S)$, it is called an *external contour* point (e.g. Q in Figure 3); when at least one of them is inside the $CH(S)$, it is called an *internal contour* point (e.g. P

in Figure 3). $VH(S)$ can also be thought of as a union of the external contour points. $IVH(S)$, on the other hand, is the union of all contour points, both external and internal, resulting a tighter fit.

The central idea of our method is to use the internal contours, *without* requiring pictures taken from the concave regions of S. First we want to point out that most internal contours are actually visible from outside, as long as one segment of the tangent line is outside $CH(S)$. For now we assume that the internal contours are already detected and present our internal contour carving algorithm below.

3.2. Locally Convex Carving

Our carving algorithm starts with an already calculated $VH(S)$ and detected internal contours. Unlike the external contours, an internal contour captured outside $CH(S)$ does not define a clear region that separates S from E^3. However, by definition, it does point to a set of points that are *locally convex*. We just do not know where they are on the tangent line. Inspired by stereo matching, we plan to use a pair of internal contours to further carve the volume. As illustrated in Figure 4, let us assume that we are given two internal contours IC_0 and IC_1 with respect to viewpoint V_0 and V_1. We intersect view rays defined by two contours to define a region R. At a first glance, R is in the concavity and should be removed (shown in Figure 4 left). But close observations lead to the conclusion that the intersecting view rays coming from the two sides of concavity may lead to over or under carving. To prevent this, we define our *Rule (1)*: if the contour normals (N_p and N_q) are opposite from each other, no carving should occur. On the other hand, if the two contours are generated from the same continuous convex surface (as shown in Figure 4 right), then R could be safely carved. However if the direction of the contour is close to parallel to the epipolar plane, the intersections cannot be reliably estimated. So we set up *Rule (2)* to prevent this: the angle between the surface normal and the epipolar plane normal should not be too small.

In summary, our *locally convex carving* is defined in Algorithm 1. The two **if** statements represents the two rules. T_o is the threshold of the angle between surface normal computed from occluding contours and epipolar plane. Also in practice we have used contours from neighboring viewpoints to do convex carving which is more reliable.

Figure 5 shows the effect with and without locally convex carving. While the area carved out may not be that significant, it actually identifies locally convex points, which are usually next to concave areas. These points are critical in our concave fitting step that will be introduced in Section 5.

4. Internal Contour Tracking

We have described our locally convex carving approach to refine the visual hull using internal contours. The problem now is how to get the internal contours on the images. While there are previous methods on contour tracking and detection (e.g., [17, 20, 29]), none of them addresses highly specular surfaces in our case. We present a semi-automatic

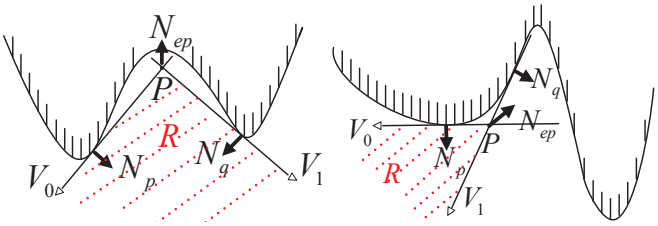


Figure 4: Locally Convex Carving. The left figure represents the case that the intersected tangent lines coming from two different convex surface and the contour normal are opposite from each other, and in this case we cannot carve out the Region R. The right one is what we defined as locally convex carving cases where the Region R can be carved. All the tangent lines in these figures indicate internal contours. N_p and N_q are the contour normals and N_{ep} is the normal for the epipolar plane.

Algorithm 1 Locally Convex Carving

for pixel p in IC_0 **do**
  find its correspondence q in IC_1 using the epipolar constraint;
  triangulate 3D point P with p and q;
  Estimate the surface normal of p and q, denoted as N_p and N_q;
  Compute the angle between N_p and N_q with epipolar plane (N_{ep}), denoted as R_p and R_q;
  if dot (cross(N_p, N_{ep}), cross (N_q, N_{ep})) < 0 **then**
    continue;
  else if $R_p < T_o$ or $R_q < T_o$ **then**
    continue;
  else
    carve out R;
  end if
end for

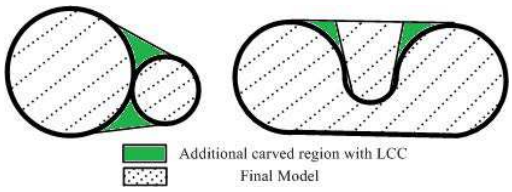


Figure 5: Locally convex carving Illustration of convex and concave cases. The figure presents the two cases that illustrate the additional regions that can be carved out with LLC and also the region that we will get after the locally convex carving operation.

approach for detecting the internal contours with contours labeled in a few key frames.

Given labeled contours from key frames, our algorithm is designed to interpolate the corresponding contours between these frames. Let us assume that we have contours C_{i-n} and C_{i+m} in Image I_{i-n} and I_{i+m} respectively, the goal is to detect the corresponding contours in Image I_i.

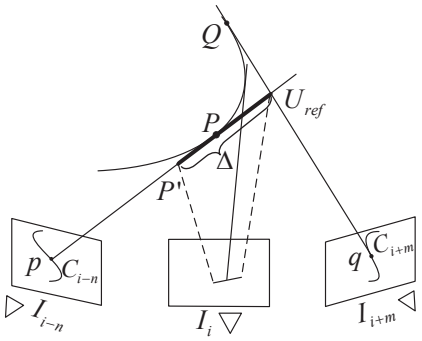


Figure 6: Geometric illustration for contour prediction. See more details in the text.

We first intersect C_{i-n} and C_{i+m} to obtain a set of 3D points, which provides a proxy about where the contour points should be in 3D. The use of the 3D proxy to guide contour detection is the main difference from typical contour tracking/detection formulations. Let p in C_{i-n} and q in C_{i+m} denote the pair of corresponding pixels and U_{ref} be the reconstructed 3D point. We also denote the contour points on the surface as P and Q. Assuming the surface between P and Q is smooth, then the possible contour points for I_i must be between P and Q. This is similar to the order constraint that is often used in stereo matching. Unfortunately we really do not know where P and Q are. However, U_{ref} can be used as loose upper bound for Q. For the lower bound, we approximate it with a user-defined parameter Δ, which defines a 3D point P' on the back-projection ray for p, but Δ distance away from U_{ref}. The line between the lower bound P' and the upper bound U_{ref} provides the searching space in 3D. Projecting P' and U_{ref} to image I_i defines the possible candidates for a contour point in I_i.

Once all the candidates for one contour are identified, we develop a global optimization approach to detect the contour points. We formulate it as a labeling problem with a data consistency term and two regularization terms. To create a uniform label set for each contour point, the 3D line segment between P' and U_{ref} is uniformly divided to a set of discrete points $\{L_k\}$, each representing a unique label.

4.1. Data term

The data term expresses the likelihood of point L_k, which projects to a particular pixel u_k in Image I_i that is a contour pixel. It is defined as the weighted sum of two sub-terms as described below.

Gradient term D_g is the gradient term, which favours the contour point to pass through pixels with strong gradient. It can be computed as:

$$D_g(k) = \exp\left(-\lambda_k G(u_k)^2\right) \tag{1}$$

$$\begin{aligned}\lambda_k = \omega_1 \mathbf{V}(u_k)\cdot\mathbf{V}(p) \\ +\omega_2 \mathbf{V}(u_k)\cdot\mathbf{V}(q)\end{aligned} \tag{2}$$

where $G(\cdot)$ is the gradient magnitude at a given pixel location and $\mathbf{V}(\cdot)$ is the gradient direction. λ_k is used to preserve the direction of the contour. It considers the differences of the gradient directions between predicted contours and two reference contours. $\omega_1 = m/(m+n)$ and $\omega_2 = n/(m+n)$ are the interpolation factors.

Histogram of intensity term D_h is the intensity matching term computed with histogram. It is assumed that the corresponding occluding contours in consecutive frames have similar color distributions (see Figure 7 for some examples). D_h is computed as follows:

$$D_h(k) = \exp\left(-H\big(I_i(u_k)\big)^2\right) \tag{3}$$

Where H is the intensity probability function computed from the intensity of pixels of C_{i-m} and C_{i+n}. In essence, we calculate a histogram of all the pixels in C_{i-m} and C_{i+n}. For u_k's intensity, we look at the corresponding bin to find its normalized probability.

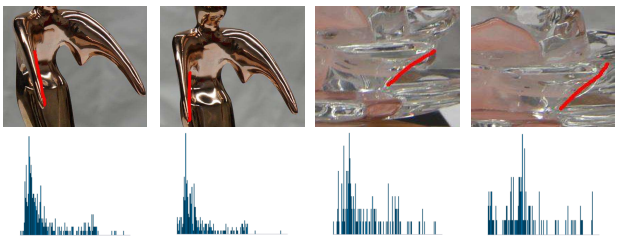


Figure 7: Intensity histogram. The upper row is captured images with internal contours highlighted in red, and the second row is the histogram of corresponding contours. Images in the first and second column are from nearby viewpoints and so as the third and fourth column.

With these terms defined, the data term for one contour point u can be defined as

$$D(u) = \sum_k (\omega_g D_g(k) + \omega_h D_h(k)) \tag{4}$$

where ω_g and ω_h are weighting factors. The data term for an entire contour, $D(C_i)$, is the sum of $D(u)$ for all u in C_i. We use the number of pixels in C_{i-n} to discretize C_i.

4.2. Regularization terms

To preserve the smoothness of the detected contour, we introduce two regularization terms. To abuse the notation a

bit, we denote a contour point in I_i as u_j, it is different from u_k, which is a candidate for one contour point. u_j has two neighbors u_{j-1} and u_{j+1}. The first term $V(u_j)$ penalizes large spatial distance of neighboring pixels.

$$V(u_j) = V(u_j|u_{j-1}) + V(u_j|u_{j+1}) \quad (5)$$

$$V(u_j|u_{j\pm1}) = 1 - exp\big(-||u_j - u_{j\pm1}||_2^2/\sigma_v^2\big) \quad (6)$$

where σ_v^2 is a normalization factor.

The second term $T(u_j)$ is to preserve the shape of C_i to be similar with reference contours C_{i-n} and C_{i+m}. We denote the corresponding contour points as p_j and q_j. We formulate in a way to preserve the Laplacian vector of u_j. Since a contour is a 1D entity, the Laplacian coordinate of u_j is calculated as $\mathbf{L}(u_j) = u_j - \frac{1}{2}(u_{j-1} + u_{j+1})$. Then $T(u_j)$ is expressed in the following:

$$T(u_j) = \Delta(\mathbf{L}(u_j), \mathbf{L}(p_j)) + \Delta(\mathbf{L}(u_j), \mathbf{L}(q_j)) \quad (7)$$

$$\Delta(\mathbf{L}(p), \mathbf{L}(q)) = 1 - \exp(-||\mathbf{L}(p) - \mathbf{L}(q)||_2^2/\sigma_t^2) \quad (8)$$

where σ_t^2 is again a normalization factor.

4.3. Energy Function

Finally we get the energy function to be minimized as,

$$E(Ci) = D(Ci) + \sum_j (\lambda_v V(u_j) + \lambda_t T(u_j)) \quad (9)$$

where λ_v, and λ_t are weighting terms. We have used the SRMP [16] to solve the high-order optimization problem with multiple labels. One contour is labeled each time.

5. Concave refinement

The locally convex carving (LCC) algorithm presented in Section 3.2 is able to reconstruct the convex part of the object surface revealed by internal occluding contours. However It cannot recover concave part since the tangent lines of a concave surface is lying inside the surface. We have developed a simple surface fitting method to estimate the concave part with some user interactions.

The basic idea is to fit a concave surface based on its boundary that can be correctly reconstructed. We first allow the user to mark a concave area. This is done in the image space. An image in which the concavity is most frontal is chosen to allow the user to mark up the concave region RC. Boundary points near RC serve as the seed points, denoted as RS. Many points in RS can be automatically identified since there is usually a transition from convex to concave, the convex part can be carved out with our LCC algorithm. A user can also include additional boundary points to give a tighter control. Figure 8 shows the concave boundary.

Then we try to propagate the boundary depth to the concave part under the smoothness constraints. We will estimate a depth value (d_p) for every pixel in RC. The energy

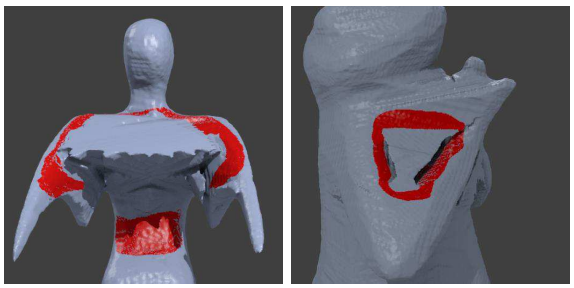


Figure 8: Concave boundary illustration. The two images illustrate the boundary cues used in concave fitting for these two models. The projection of these 3D boundary vertices is RV aera in the frontal 2d image.

function that needs to be minimized can be expressed as follows:

$$\begin{aligned} E_d = \lambda_c \sum_{p \in RV} ||d_p - \tilde{d}_p||_2^2 + \lambda_{r1} \sum_{\substack{p,q \in RC \\ q \in N(p)}} ||d_p - d_q||_2^2 \\ + \lambda_{r2} \sum_{\substack{p,q \in RC \\ q \in N(p)}} ||\delta d_p - \delta d_q||_2^2 \end{aligned} \quad (10)$$

where λ_c, λ_{r1} and λ_{r2} are the weights for corresponding terms and $N(p)$ defines the four neighbors of pixel p. The first data term measures the depth difference between the known point $\tilde{d}_p$ and the depth value of the reconstructed point d_p. The second and third term enforce the smoothness constraints with δd_p stands for the depth gradient.

The above energy function can be formulated as a linear least square system for which the global optimum can be efficiently computed. Once we get depth map for RC, we will use it to carve out any volume that is in the concave part of the reconstructed surface. That leads to the final model.

6. Experiments and Results

We evaluate the proposed method on four challenging objects, consisting of shiny specular objects (*statue, trophy*, and *frog*) and one transparent object made of glass (*lotus*). Forty-five $2000 * 3000$ images were captured for each object when it was placed on a checkerboard pattern (for pose calculation). The object silhouettes were extracted using Grab-cut.

6.1. Contour Tracking Results

We have verified our contour tracking method on the four datasets. Several contours were scribbled first in a set of key frames. Based on the complexity of objects, generally we need to label internal contours in every 5-7 consecutive images (see supplementary materials). From these labelled contours on the key frames, we ran our contour tracking algorithm. The parameters were set to $\Delta = 20mm$, $w_g =$

$w_h = 0.8$, $\sigma_v = 2.0$, $\sigma_t = 2.5$, and $\lambda_v = 1.2, \lambda_t = 1.0$. These values were tuned empirically and remained fixed for all four data sets.

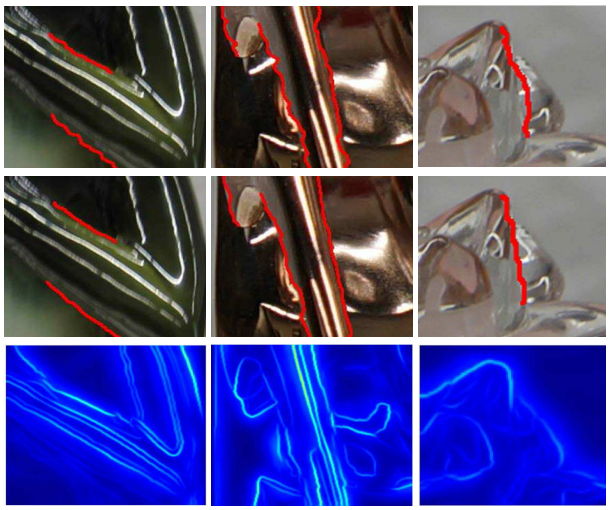


Figure 9: Comparison of contour tracking. The detected pixels are highlighted in images. The first row shows the results using only the gradient data term and the pairwise smoothness term. The second row shows the results with all the data terms and regularization terms. The third shows the general contour detection results using gPb algorithm [1].

Figure 9 shows some qualitative comparisons with gradient + smoothness only method. As we can see in the first row, for *statue* and *lotus* (right two) the detected contours can be snapped to highlights where the gradient is really strong. With our proposed terms, we can still get good results in these cases, as shown in the second row. For *frog* (left), comparable results are archived since it is not as specular as the other two. The general contour detection method used in [29] is not suitable for our case as shown in the third row, where the highlight areas have large probability to be detected as contours. This also indicates why previous contour tracking methods are likely to fail since they would also be confused by the gradient caused by highlights.

Quantitative Evaluation We further manually labelled all the images to quantify the accuracy of our tracking results. We calculate the mean distance (in pixels) between the tracked contour and labelled data, which are considered as ground truth. The results are shown in Table 1. It shows the effect of different terms. With only the gradient term we cannot get pleasant results especially for statue and lotus. As we integrate the proposed data terms and regularization terms, the mean pixel error has been reduced to half.

6.2. Reconstruction Results

We present our final reconstructed 3D models in Figure 10 with comparison to visual hull reconstruction. The

	Statue	Lotus	Trophy	Frog
G	2.3178	2.3516	1.9448	1.7657
G+IH	1.3902	1.6106	1.2256	1.1423
G+IH+P	1.2503	1.3961	1.1028	0.9869
G+IH+P+T	**1.1221**	**1.2324**	**1.0509**	**0.9381**

Table 1: Mean error of tracked contours. The table gives the mean pixel error on four datasets and the rows indicate the terms that were incorporated. G donates the gradient term data (eq. 1) and IH is the term using histogram of color intensity (eq. 3); P and T are two regularization terms of eq. 5 and eq. 7 respectively.

threshold T_o in the LCC algorithm is chosen from 30 to 45 degrees. As for the weighting parameters for each term in the fitting formulation, we use $\lambda_c = 0.5$, $\lambda_{r1} = 0.2$, and $\lambda_{r2} = 0.3$ for the four models. It will take about twenty mininutes to get the 3D model with user gudiance.

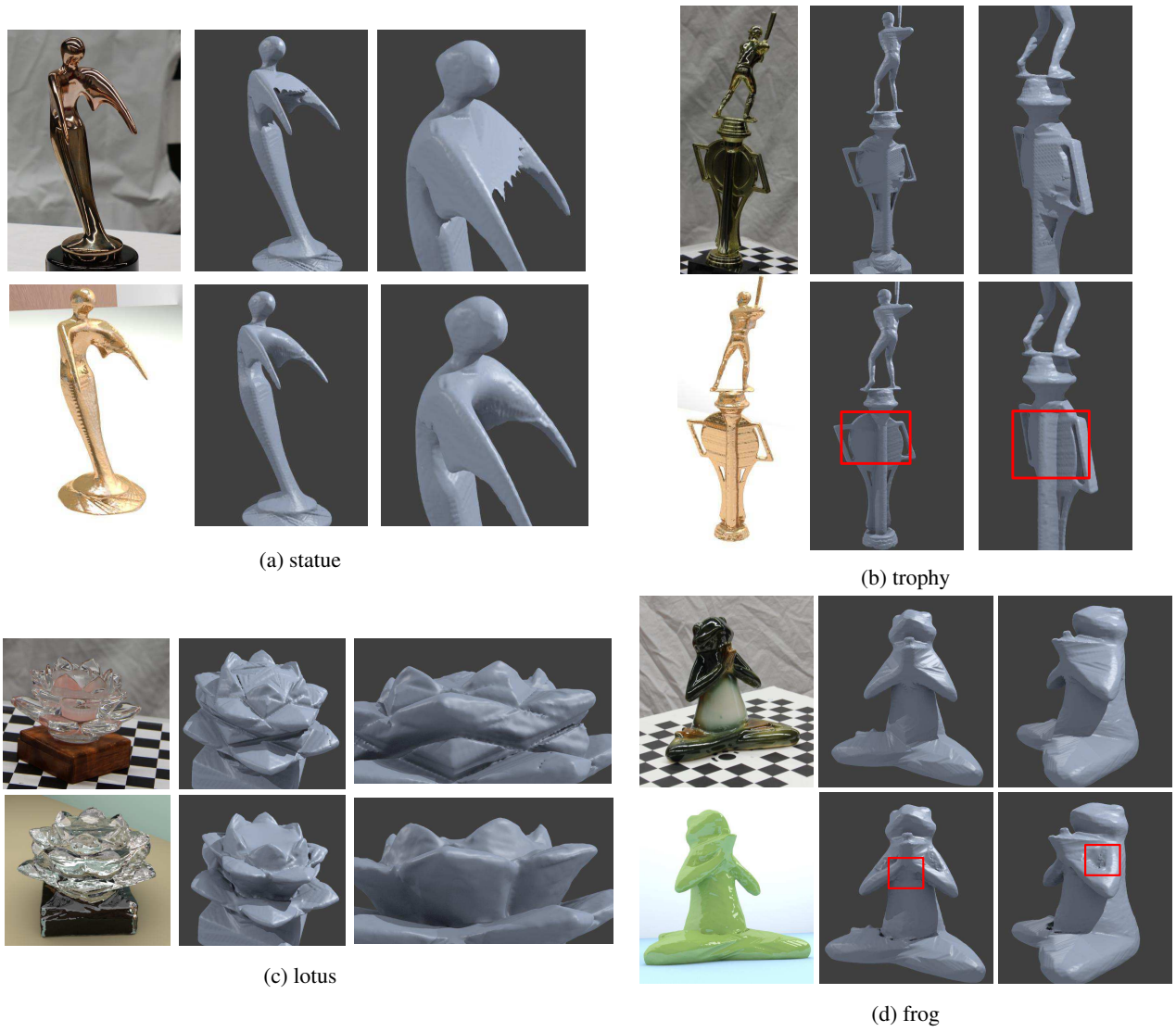
As shown in Figure 10a, the reconstructed statue model is much closer to the real surface than the original viusal hull model. The left/right wings of the model are refined using our LCC method with detected contours along the wings. And then concave fitting is performed on the back of the statue to have smooth transition with its surroundings.

For the lotus model (Figure 10c), it has three layers of petals, which are barely recognizable from the visual hull model, while the geometry has got revealled successfully with our convex carving procedure. The concave part on the top of the model is fitted to have smooth transition with the petals on the top layer.

For the frog model (Figure 10d), we are able to carve out the locally convex part along the left/right arms and also the belly under the hands, which provides the boundary and gradient propagation cues for concave fitting, as shown in Figure 8. Therefore, the fitted surface has preserved the tendency of the surface.

For the trophy model (Figure 10b), the pillar of the model is completely reconstructed with our method, as marked red on the model. No concave fitting is performed on this model.

Limitations There are still some limitations in our method. Our concave fitting method tends to under fit concave areas. A better user interaction method is needed, such as push of surfaces. Contour tracking on highly specular objects are still very challenging, in particular for areas with small details, such as the face of the baseball player on the trophy. These are difficult for even our human eye to see. Overall our method is better suited for reconstruction of organic shapes without detailed surface relief patterns. Finally image segmentation on glass objects is very difficult, even with user interactions. It probably requires a more controlled setup.



(a) statue

(b) trophy

(c) lotus

(d) frog

Figure 10: Comparison of reconstructed 3d models. For each object, the first row shows the original image from one viewpoint and reconstructed visual hull surface, and the second row shows the rendered and reconstructed model with our proposed method.

7. Conclusion

This paper addressed the problem of multi-view surface reconstruction for non-Lambertian objects. Instead of using active lights or other specialized setup, we aim to use multi-view stereo capture setup under general unknown illumination. Our visual-hull-based reconstruction approach exploits internal contours inside the objects silhouettes to generate a tighter surface model. Given the internal contours, our novel locally convex carving algorithm is able to carve out extra voxels in convex part of the surface. Also, this carving operation provides important boundary cues for fitting concave areas that do not have any contours. Our second contribution is a novel approach for tracking internal contours given contours labeled in sparse key frames. It is designed specifically for highly specular or transparent objects, for which assumptions made in traditional contour detectiong/tracking methods are no longer valid. We have validated our methods, both quantitatively and qualitatively, with four datasets of different object materials. Results show that we are able to generate visually pleasing models for very challenging cases.

Acknowledegments The work is supported in part by US National Science Foundation grant IIS-1231545, National High Technology Research and Development Program("863"Program) of China, Key Industrial Innovation Chain of Shaanxi Province Industrial Area (2015KTZDGY04-01) and Chinese Scholarship Council. Ruigang Yang is the corresponding author for this paper.

References

[1] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):898–916, 2011. 7

[2] S. Avidan. Support vector tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(8):1064–1072, 2004. 2

[3] E. Boyer and M. O. Berger. 3d surface reconstruction using occluding contours. *International Journal of Computer Vision*, 22(3):219–233, 1997. 2

[4] E. Boyer and J. S. Franco. A hybrid approach for computing visual hulls of complex objects. In *CVPR*, pages 695–701, 2003. 2

[5] M. Brand, K. Kang, and D. B. Cooper. Algebraic solution for the visual hull. In *CVPR*, pages 30–35, 2004. 2

[6] V. Chari, A. Agrawal, Y. Taguchi, and S. Ramalingam. Convex bricks: A new primitive for visual hull modeling and reconstruction. In *ICRA*, pages 770–777, 2012. 2

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

[8] Y. Ding, F. Li, Y. Ji, and J. Yu. Dynamic fluid surface acquisition using a camera array. In *ICCV*, 2011. 1, 3

[9] J. Franco and E. Boyer. Efficient polyhedral modeling from silhouettes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(3):414–427, 2009. 2

[10] Y. Furukawa and J. Ponce. Carved visual hulls for image-based modeling. In *ECCV*, 2006. 2

[11] P. Gargallo, E. Prados, and P. Sturm. Minimizing the reprojection error in surface reconstruction from images. In *ICCV*, 2007. 2

[12] A. Hertzmann and S. M. Seitz. Example-based photometric stereo: Shape reconstruction with general, varying brdfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1254–1264, 2005. 1

[13] V. H. Hiep, R. Keriven, P. Labatut, and J. P. Pons. Towards high-resolution large-scale multi-view stereo. In *CVPR*, pages 1430–1437, 2009. 2

[14] W. Hu, X. Zhou, W. Li, W. Luo, X. Zhang, and S. Maybank. Active contour-based visual tracking by integrating colors, shapes, and motions. *IEEE Transactions on Image Processing*, 22(5):1778–1792, 2005. 2

[15] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988. 2

[16] V. Kolmogorov. A new look at reweighted message passing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(5):919–930, 2015. 6

[17] K. N. Kutulakos and C. R. Dyer. Occluding contour detection using affine invariants and purposive viewpoint control. In *CVPR*, pages 356–361, 1994. 2, 4

[18] A. Laurentini. The visual hull of curved objects. In *ICCV*, pages 356–361, 1999. 2, 3

[19] S. Lazebnik, Y. Furukawa, and J. Ponce. Projective visual hulls. *International Journal of Computer Vision*, 74(2):137–165, 2007. 2

[20] G. Li, Y. Tsin, and Y. Genc. Exploiting occluding contours for real-time 3d tracking: A unified approach. In *ICCV*, 2007. 2, 4

[21] D. Liu, X. Chen, and Y. H. Yang. Frequency-based 3d reconstruction of transparent and specular objects. In *CVPR*, pages 660–667, 2014. 1, 3

[22] M. Liu, K. Y. Wong, Z. Dai, and Z. Chen. Pose estimation from reflections for specular surface recovery. In *ICCV*, 2011. 3

[23] S. P. Mallick, T. E. Zickler, D. Kriegman, and P. N. Belhumeur. Beyond lambert: Reconstructing specular surfaces using color. In *CVPR*, pages 619–626, 2005. 3

[24] A. Mondal, S. Ghosh, and A. Ghosh. Efficient silhouette-based contour tracking using local information. *Soft Computing*, pages 1–21, 2004. 2

[25] D. Nehab, T. Weyrich, and S. Rusinkiewicz. Dense 3d reconstruction from specularity consistency. In *CVPR*, pages 1–8, 2008. 1

[26] N. Paragios and R. Deriche. Geodesic active regions and level set methods for motion estimation and tracking. *Computer Vision and Image Understanding*, 97(3):259–282, 2005. 2

[27] N. Ray and S. T. Acton. Motion gradient vector flow: An external force for tracking rolling leukocytes with shape and size constrained active contours. *IEEE Transactions on Medical Imaging*, 23(12):1466–1478, 2004. 2

[28] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics*, 23(3):309–314, 2004. 2

[29] Q. Shan, B. Curless, Y. Furukawa, C. Hernandez, and S. M. Seitz. Occluding contours for multi-view stereo. In *CVPR*, pages 4002–4009, 2014. 2, 4, 7

[30] R. Szeliski and R. Weiss. Robust shape recovery from occluding contours using a linear smoother. *International Journal of Computer Vision*, 28(1):27–44, 1998. 2

[31] M. Weinmann, A. Osep, R. Ruiters, and R. Klein. Multi-view normal field integration for 3d reconstruction of mirroring objects. In *ICCV*, 2013. 1, 3

[32] T. E. Zickler, P. N. Belhumeur, and D. J. Kriegman. Helmholtz stereopsis: Exploiting reciprocity for surface reconstruction. *International Journal of Computer Vision*, 49(2-3):215–227, 2002. 3