Interactive High-Relief Reconstruction for Organic and Double-Sided Objects from a Photo

Chih-Kuo Yeh, Shi-Yang Huang, Pradeep Kumar Jayaraman, Chi-Wing Fu, *Member, IEEE*, and Tong-Yee Lee, *Senior Member, IEEE*

Abstract—We introduce an interactive user-driven method to reconstruct high-relief 3D geometry from a single photo. Particularly, we consider two novel but challenging reconstruction issues: i) common non-rigid objects whose shapes are organic rather than polyhedral/symmetric, and ii) double-sided structures, where front and back sides of some curvy object parts are revealed simultaneously on image. To address these issues, we develop a three-stage computational pipeline. First, we construct a 2.5D model from the input image by user-driven segmentation, automatic layering, and region completion, handling three common types of occlusion. Second, users can interactively mark-up slope and curvature cues on the image to guide our constrained optimization model to inflate and lift up the image layers. We provide real-time preview of the inflated geometry to allow interactive editing. Third, we stitch and optimize the inflated layers to produce a high-relief 3D model. Compared to previous work, we can generate high-relief geometry with large viewing angles, handle complex organic objects with multiple occluded regions and varying shape profiles, and reconstruct objects with double-sided structures. Lastly, we demonstrate the applicability of our method on a wide variety of input images with human, animals, flowers, etc.

Index Terms—Reconstruction, high-relief, lenticular posters, single image, folded, double-sided, object modeling, depth cues, completion, inflation

1 Introduction

Given the vast availability of 2D images that can be easily obtained from the Internet or photo taking, say, with a cell phone, we aim to develop in this work a novel interactive solution to support the reconstruction of high-relief geometry from a single photo. In particular, we focus on high-relief geometric reconstruction of non-polyhedral shapes (see Figs. 1 and 2 for various examples). High-relief conveys much more depth information as compared to bas-relief; it requires a significant mass of the figure object to be elevated from the background plane [1], while some parts may be completely detached from the background [2].

This would be useful for supporting applications such as stereoscopy and rotating lenticular posters. Note that rotating lenticular posters (used as advertisements in "The Hobbit" and "The Amazing Spiderman") feature objects that appear to be raised from the background like a relief sculpture and rotate when we look around it, thereby delivering an interesting out-of-the-poster 3D perception.

- *C.-K. Yeh, S.-Y. Huang, and T.-Y. Lee are with the Department of Computer Science and Information Engineering, National Cheng-Kung University, Tainan City 701, Taiwan, R.O.C.*
  E-mail: {simpson.ycg, t8590338}@gmail.com, tonylee@mail.ncku.edu.tw.
- *P.K. Jayaraman is with the School of Computer Engineering, Nanyang Technological University, Singapore 639798, Singapore.*
  E-mail: pradeep.pyro@gmail.com.
- *C.-W. Fu is with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong.*
  E-mail: philip.chiwing.fu@gmail.com.

Manuscript received 14 Aug. 2015; revised 10 May 2016; accepted 22 May 2016. Date of publication 1 June 2016; date of current version 26 May 2017.
Recommended for acceptance by T. Igarashi.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TVCG.2016.2574705

Compared to conventional methods for single-photo 3D reconstruction, this work has the following distinctive aspects. First, we aim to support high-relief rather than bas-relief 3D reconstruction, so that the geometry can appear to pop out of the image plane when we look around it. Existing methods that hallucinate stereoscopic viewing [3], [4] or reconstruct shallow geometries [5] may not be able to achieve this. Second, we aim to support the reconstruction of common non-rigid objects such as human, animals, flowers and teapots. Here, the shapes of the objects can be organic rather than polyhedral or symmetric, and different sides (front and back) of some curvy parts may be revealed simultaneously in the image. Conventional 3D reconstruction methods are insufficient to handle these scenarios.

Sketch-based modeling [6], [7] is another research topic relevant to this work. Their methods provide fine-grained controls for users to author a full 3D model. However, they often require the users to specify tedious amount of constraints for creating more complex shapes. In this work, we also take an interactive user-driven approach, but we aim for reconstructing geometry from photos with a few markup cues from the users, where the user provided information is sufficient for supporting applications such as stereoscopy and rotating lenticular posters.

To achieve the research goal is highly challenging since we only have a single view of the objects in the input photo. More specifically, the object to be reconstructed could be organic in shape (see Fig. 1(left)). Moreover, it may involve multiple depth layers (hereafter referred to as regions) (see Fig. 1(middle)); each region could have varying geometric characteristics: convex, concave, or a mixture of both, and some regions could be partially occluded by others.

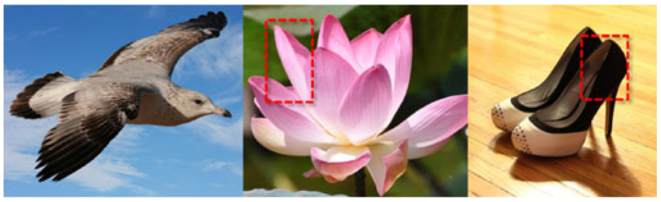


Fig. 1. Examples of organic objects and double-sided structures (flower petal and shoe opening) that can be handled by our method.

Previous methods [3], [4] consider individual objects in the image as separated regions, and do not explicitly model their surface characteristics. Furthermore, we may encounter objects with front and back sides in the given photo view, e.g., flower petals, shoes, etc. (see Fig. 1(middle and right)), resulting in view-dependent self-occlusions; no previous work has considered this double-sided reconstruction scenario.

We address the above issues by a user-driven approach. Here, we start with a user-driven segmentation of the input image. Then, we automatically infer local layering information and construct a 2.5D model by analyzing T-junctions and overlap among the adjacent regions. After that, we complete the missing portions of each region with the help of the local layering information, where we consider three types of completion cases corresponding to common occlusion situations in natural images. Next, users can interactively mark up slope and curvature cues in our interface to guide our constrained optimization model to inflate the surface geometry; real-time preview of the inflation results is provided. Compared to previous inflation methods, which mainly produce simple convex surfaces, our method can handle much more complex shapes. Lastly, we arrange the regions in 3D and stitch them together to avoid visible gaps and discontinuities; hence, we can produce plausible high-relief results and allow much larger viewing angles.

Our method has several advantages over existing 3D reconstruction methods. First, it can deal with common everyday objects without specific assumption on the object shape, e.g., similarity [8], planarity [9], and geometry [10]. Second, compared to [5], which produces only shallow geometry, our method can produce high-relief models, allowing the viewing of the reconstructed model at angles as large as 45 degrees from the image normal. Third, it does not require any external database for model fitting [11]. Lastly, our inflation method, which takes only simple user inputs, can support the reconstruction of complex 3D shapes with parts that are convex, concave, or even a mixture of both. These are demonstrated with wide variety of results produced from different kinds of input images.

2 Related Work

Single-view 3D Reconstruction is a very challenging problem. It is highly under-constrained, and requires extensive amount of information in order to achieve plausible reconstruction results. In general, there are three major approaches: fully automatic, user-driven, and semi-automatic. Since there is a huge volume of previous works in this area, we discuss only the closely related ones below.

Fully automatic methods such as [3], [12], [13] reconstructed an outdoor 3D scene by first decomposing the scene into ground, sky, and vertical objects. Similarly, Saxena et al. [9] extracted the 3D structure of outdoor scenes by combining local image features, connectivity, coplanarity, and collinearity in a probabilistic model. More recently, Zeng et al. [4] created proxy 3D models using generalized cylinders from segmented input images. However, the segmented regions in their inputs correspond to individual objects, and there is no depth disparity among parts within the same object. Hence, the method may not be able to handle the reconstruction of more complex objects. Shape-from-shading [14] is another closely-related area, but this approach may not be suitable for images with prominent textures, since this would interfere with the analysis of the shading. In general, fully automatic methods are insufficient to support the reconstruction of complex objects due to the lack of 3D information available in a single image.

User-driven methods are another approach for constructing 3D geometry information; typically, it can be used with or without a fixed reference image. Among the methods without a fixed reference image, Igarashi et al. [15] developed a sketch interface to inflate 3D shapes from user-input 2D contours. Karpenko and Hughes [16] devised a rigorous method to reconstruct a fairly smooth surface not only from the contours, but also suggestive contours along the silhouette. Ijiri et al. [17] exploited domain-specific knowledge to create complex shapes such as flowers. More recently, Nealen et al. [6] employed different forms of control curves to iteratively cut, extrude, and tunnel a 3D model, enabling the creation of more complex shapes. Since these methods are generally used for authoring 3D contents, they are not designed for 3D modeling with a fixed reference image.

Among the user-driven methods with a fixed reference image as an input, Van Overveld [18] used gradients to help reconstruct the 3D geometry. Oh et al. [19] divided the input image into layers, manually inpainted the hidden parts, and then used several tools, e.g., vertical tool and push/pull tool, to create an architectural model that corresponds to the input image. Similar to our work, Zhang et al. [20] allowed users to put in sparse orientation constraints on input image

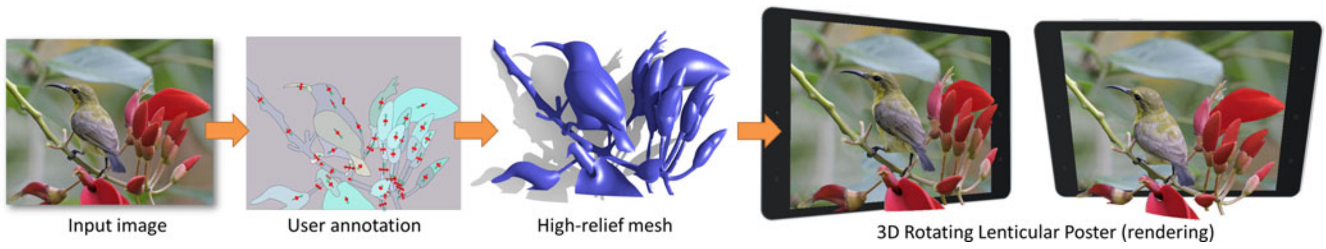


Fig. 2. From a single input image, our method constructs a high-relief 3D model by 2.5D layering, layer completion, inflation, and stitching with only simple user-provided cues. Our result can be rotated to produce plausible renderings from large viewing angles.

to produce a smooth 3D surface. However, the methods above cannot generate folded and double-sided structures as in this paper. Later, Cordier et al. [21] reconstructed 3D structures from 2D sketches that illustrate objects with partial- or self-occlusion, e.g., a torus knot. In contrast, our method not only can generate tubular structures, but also folded and double-sided structures in 3D.

Other than gradient, Wu et al. [22] proposed to interactively transfer normals from a known 3D primitive as a palette to the input image for 3D reconstruction, while Joshi and Carr [7] proposed to put sparse constraints along sketch lines to indicate surface smoothness and depth discontinuities. Similarly, we also use sparse constraints, but the main difference is that our method can generate geometric structures with multiple layers rather than just a single layer, and appropriately connect the geometry across layers. Gingold et al. [23] proposed another approach to refine a 3D design by manipulating user-annotated primitives, e.g., tilt handles and cross-sectional markups. However, there are limited number of primitives and the method cannot produce flexible shapes with holes and folded structures, as demonstrated in our results. Unlike the above works, our method can (semi-)automatically reconstruct surface geometry via information extracted from the reference image.

Semi automatic approach combines user inputs and prior knowledge from images to enhance the reconstruction quality and minimize user effort, e.g., Prasad et al. [24], [25] reconstructed smooth 3D surfaces that project on user-defined silhouettes extracted from the 2D image. Several classes of objects have been employed as proxies for 3D reconstruction, e.g., cuboid [26], cylindrical [10], [23], and conical proxies specifically for flower petals [8]. Additionally, knowledge from a stock of 3D models can help recover the geometry and texture information from 2D images [11]. While applicable to a wide variety of everyday objects, the above methods cannot plausibly handle complex and organic shapes such as human bodies and animals.

The method proposed by Sýkora et al. [5] allows efficient reconstruction of 3D bas-relief meshes from 2D hand-drawn cartoons by using some user-specified depth cues. While the method enables 3D-like enhancements such as global illumination, the reconstructed mesh is relatively flat and does not have sufficient 3D structure to allow larger viewing angles as compared to our case.

Our method overcomes the issues above in two key aspects. First, we propose a novel completion framework that considers three different completion cases; in particular, we consider double-sided structures by appropriately connecting and completing the associated front and back sides of the same surface into a folded structure. This cannot be achieved by any previous work. Second, we develop an interactive user-driven method that takes very little user inputs (slope and curvature cues) to reconstruct 3D geometry via a constrained optimization model. Further with our inflation and stitching methods, we can produce plausible high-relief geometry for complex, organic, and double-sided structures, such as those shown in the paper.

2.5D Modeling. Rather than full 3D reconstruction, an alternative approach is to represent the segmented regions in the input image as planar layers and model the occlusion among the layers, e.g., Horry et al. [27] extracted the 2.5D structure in a natural image by estimating the vanishing point and perspective projection of a scene and then approximating the foreground and background objects with planar layers. Rivers et al. [28] combined several views of an object in the form of vector drawings to create a rotatable 2.5D model. More recently, Yeh et al. [29] reconstructed a 2.5D hair model from a single cartoon image to support hair animations and manipulation. Other methods related to 2.5D reconstruction typically work with videos instead of images, e.g., [30], [31]. Overall, the above works do not explicitly recover 3D geometric information about the objects in the input image, so they cannot simulate realistic 3D rotations as required in our problem.

3 Overview

Fig. 3 illustrates the procedure of our method with a running example. Overall, there are two major components:

3.1 2.5D Modeling

Segmentation. This step is an important prerequisite to partition an image into semantic regions. This has been an active research area, but obtaining accurate results automatically is still extremely challenging due to varying perception of ground-truth, even among humans. Here, we extend the standard pixel-based graph-cut method [32] to be (2D) mesh-based to produce sharper segmentation boundary: the user scribbles foreground and background labels on the input image to iteratively subdivide it (see Fig. 3b (top)) into semantic *regions*, where each region consists of pixels that are perceived to be depth continuous.

Local layering. Next, we determine and categorize the local layering, or depth ordering, between every pair of neighboring segmented regions: *above*, *below*, or *equal depth* (see the orange arrows in Fig. 3b (top) that reveal the order). This step is necessary not only to provide a perception of depth, but also to avoid intersections when we inflate the regions later. Here, we employ the hypotheses in [29], [31], i.e., T-junctions and overlap area, to construct the local layering information. For detail, please refer to these papers.

Completion. To seamlessly rotate a reconstructed result (high-relief model), we not only have to reconstruct the local geometry of each layer (or region), but also have to complete the occluded parts in each region, so that holes will not appear around the occlusion when we rotate the resulting models. Here, we consider three different completion cases (see Section 4): i) expand a single region from the occluding boundary; ii) connect non-neighboring regions under a common occlusion; and iii) double-sided structure, where portions of its front and back sides are revealed simultaneously in the given image. Note that this step has to complete both geometry and texture, while considering the depth ordering among the regions (Fig. 3b(bottom)).

3.2 High-relief Modeling

Inflation. To reconstruct the depth information on each completed region, we take only simple user cues as hints (see Section 5.1): slope cues (Fig. 3c(top)) and curvature cues (Fig. 3c (middle)). Further, we formulate an optimization model to efficiently inflate each region while respecting the constraints (see Section 5.2). In particular, our method has two main

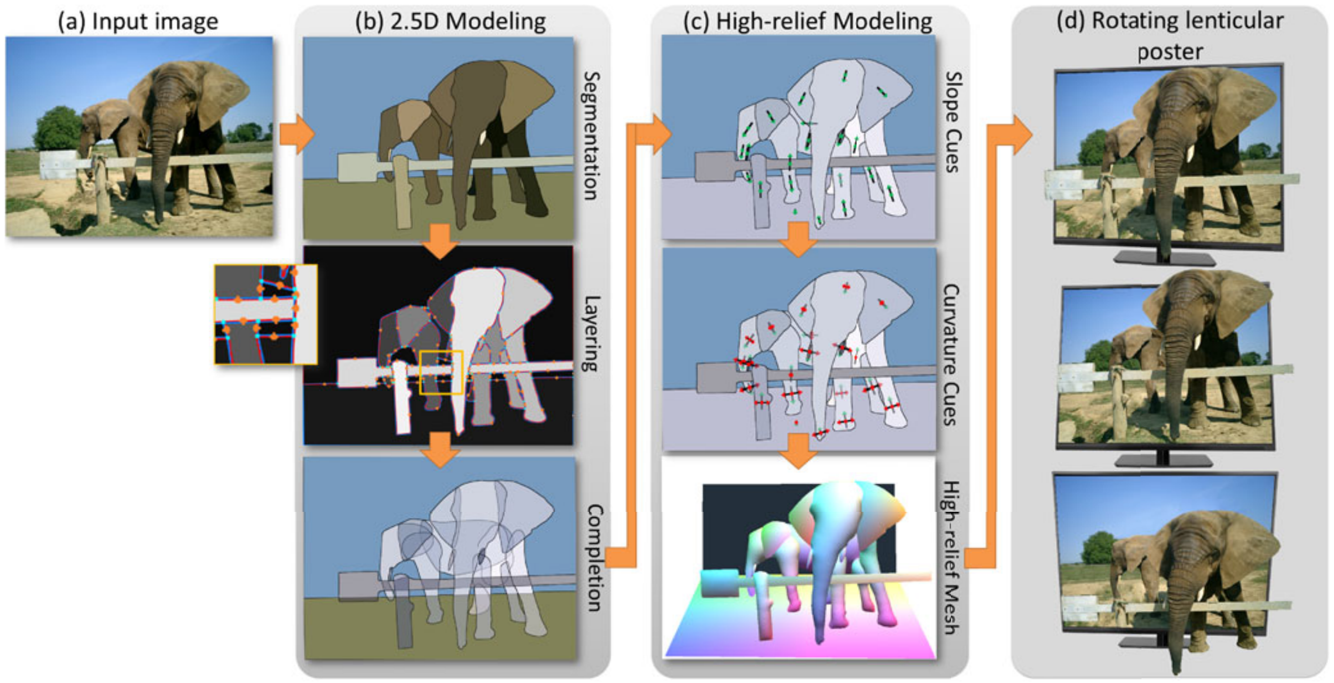


Fig. 3. Overview of our approach. Given an input image (a), we first generate a 2.5D model (b) by user-driven segmentation, automatic layering, and region completion. Then, using the slope and curvature cues marked up by users, we can lift up the planar regions to 3D by inflation and stitching, and produce a high-relief 3D model (c). Such a result can be plausibly rotated to produce the example rotating lenticular poster (renderings) (d).

advantages compared to previous user-driven methods. First, it does not require tedious specification of many different constraints while rotating the model during the reconstruction (e.g., [6]). Second, compared to state-of-the-art inflation methods such as [5], our method enables slanting of layers (by the slope cue) and control over the shape profile (by the curvature cue). By this, we can interactively inflate a wide range of natural objects with nontrivial organic shapes.

Stitching. Lastly, we sort and arrange the inflated regions based on the layering information and stitch adjacent inflated regions accordingly, so that gaps between regions do not appear when we rotate the model (Fig. 3c(bottom)). In particular, we formulate another optimization model to minimize the gaps in-between regions while maintaining proper depth ordering and connecting adjacent regions along their shared boundary with minimum deformation.

The forthcoming sections present key techniques we developed in this work: Section 4 for the completion step, Section 5 for the inflation step, and Section 6 for stitching.

4 Completion

The goal of completion is to appropriately fill the holes related to the occluded regions, so that we can produce plausible high-relief models that can be rotated, see Fig. 4. However, this is especially challenging with a single image as input, since we have to reconstruct the hidden occluded parts without any prior knowledge. In detail, we identify and consider three types of completion cases:

Case (i): Expand a single region from the occluding boundary. This case refers to occluding boundaries that are not classified as the next two cases, i.e., an individual region is simply partially occluded by its neighboring regions. E.g., see the top occluding boundary of the right elephant's rightmost leg in Fig. 3b(mid), it is partially occluded by the elephant's right ear, so we extend the leg region upward from the boundary by a level-set method (extended from [33]). This method iteratively pushes the boundary curve from the boundary to expand the related region until the expanded boundary is smooth at the occluding endpoints (see Fig. 3b(bottom) for the result), while ensuring that the boundary curve is always below the associated neighboring regions.

Case (ii): Connect non-neighboring regions under a common occlusion. Here, an occlusion breaks an object part into multiple non-neighboring regions, e.g., see Fig. 3b (top): the long horizontal bar of the fence is occluded by the right elephant's trunk. To resolve this case, we examine every pair of occluding boundaries among the segmented regions and look for a matching pair (under a common occluder) by the following criteria: i) RGB color histogram distance (d_c) by computing normalized cross-correlation between histograms with 256 bins, ii) distance between corresponding endpoints (d_e), where we normalize the image width to be 1.0 with fixed aspect ratio, and iii) difference in tangent-vector angles (in radian) at the T-junction endpoints on the occluding boundaries (d_t). If the joint probability $e^{-\sigma_c d_c^2} \cdot e^{-\sigma_e d_e^2} \cdot e^{-\sigma_t d_t^2} > 0.8$,

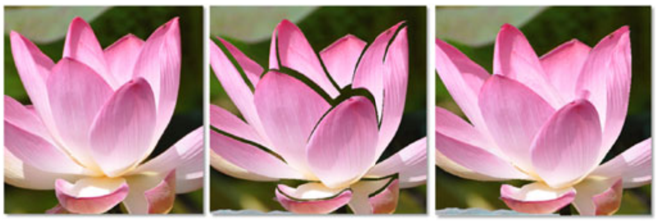


Fig. 4. Left: input image. Middle: rotate the high-relief result reconstructed with inflation and stitching but not completion; undesired holes will appear. Right: same result but reconstructed also with completion.

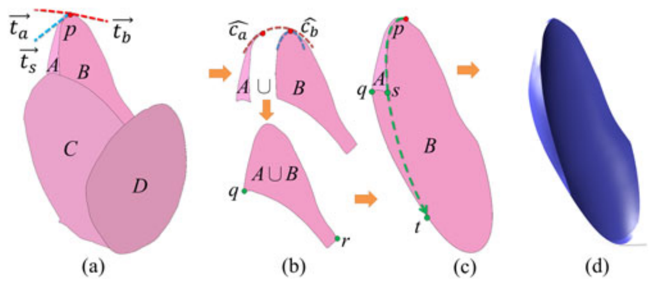


Fig. 5. Completion procedure in case (iii-a) for the top two flower petals in Fig. 4. (a) two neighboring regions A and B are identified as Case (iii-a) by analyzing the associated tangents vectors and turning angles at boundary end-points, e.g., p; (b) we extend B to be $A \cup B$, (c) and use a co-completion strategy to complete the occluded portion of A and new B under C and D. (d) Preview: the reconstructed petal (A and B) in 3D.

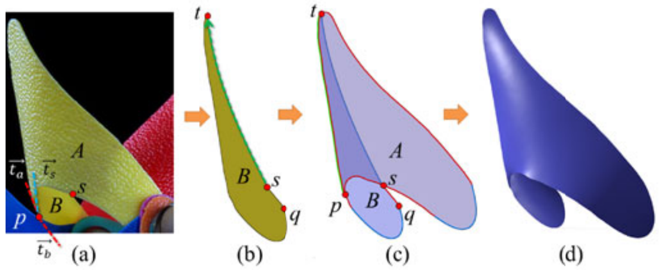


Fig. 7. Completion procedure in case (iii-b) with the pinwheel shown in Fig. 6. (a) two neighboring regions A and B that are identified as Case (iii-b); (b) complete B by the procedure described below in case (iii)(b). (c) the completed result, and (d) Preview: the reconstructed 3D pinwheel region (A and B) after the upcoming inflation and stitching steps.

we regard this as a match, e.g., the fence case in Fig. 3b. Here, $\sigma_c = 1.0$, $\sigma_e = 2.0$, and $\sigma_t = 0.125$ for all results produced in the paper. After we find a match, we construct two Bézier curves (with tangent magnitude set to $0.3d_e$) to connect the related endpoints with C^1 continuity, and combine the two regions into one, see Fig. 3b(bottom) for the result of the fence. Additionally, the user may manually connect regions together.

Case (iii): Double-sided structures. This case commonly arises in objects with thin and curved structures, where both front (A) and back (B) sides of the same surface are exposed. Here, we consider two sub-cases: (a) fully-covered, where B can be extended fully to be $B \cup A$, e.g., see the top two flower petals in Fig. 5 and the teapot opening in Fig. 6 and (b) partially-covered, where B can only be extended to be $B \cup$ partial A, e.g., pinwheel and ribbon cables (see Figs. 6 and 7). Overall, we aim to identify the front-and-back association and complete the related regions plausibly into a double-sided geometry with the help of the inflation step.

To identify case (iii), we examine every 3-valence junction formed between a pair of neighboring regions, e.g., point p in Fig. 5a. Here, we denote A and B as two related neighboring regions, where A is above B (without loss of generality); note that this information was produced earlier from the layering step. Next, we compute tangent vectors $\vec{t_a}$, $\vec{t_b}$, and $\vec{t_s}$ at point p along A's outer boundary, B's outer boundary, and their shared boundary, respectively (see Fig. 5a), and the angles in-between $\vec{t_a}$ and $\vec{t_s}$ (denoted as θ_a), and $\vec{t_s}$ and $\vec{t_b}$ (denoted as θ_b). Around p, we also determine two boundary curves $\widehat{c_a}$ and $\widehat{c_b}$ (see Fig. 5b).

If $\theta_a + \theta_b \leq \pi$ and $\widehat{c_a}$ and $\widehat{c_b}$ are C^1 continuous, we regard A and B as a candidate pair for Case (iii), since by observation, their combined region locally forms a convex shape, e.g., $A \cup B$ as the back and A as the front (see Fig. 5b). Moreover, these conditions are satisfied only when the shared boundary of A and B is completely enclosed within $A \cup B$; this helps to filter out cases of non-double-sided occluding regions. We repeat the above procedure for all pairs of neighboring (occluding) regions and present the resulting candidate pairs to user for selection.

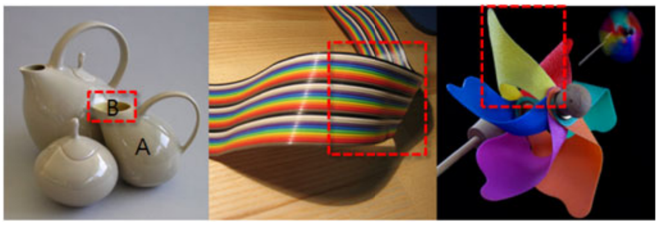


Fig. 6. More examples of double-sided structures (see also Fig. 1), where we see different sides of the same surface: the teapot opening (case iii-a: fully-covered, where B can be extended to $B \cup A$) and cable and pinwheel structures (case iii-b: partially-covered), highlighted in red.

For neighboring regions identified as Case (iii), we complete their occluded parts while respecting their local layering by the following co-completion strategy:

(iii-a) Fully-covered: we first extend B to be $A \cup B$ (see Fig. 5b) since A (front) intuitively occludes part of B (back). Then, we use the procedure in Case (i) to expand B towards C and D from the boundary curve $\widehat{qr}$ (see Figs. 5b and 5c), and then complete A by smoothly extrapolating its non-shared boundary $\widehat{ps}$ to find $\widehat{st}$ (see Fig. 5c), where t is on the boundary of completed B, so the occluded part of A under C is bounded by $\widehat{qt}$ and $\widehat{st}$. Lastly, we connect A and B along the outer boundary given by $\widehat{pqt}$.

(iii-b) Partially-covered: we complete the occluded portion of B (bounded by p, s, and t) by smoothly extrapolating $\widehat{qs}$ until we find t, which is on A's boundary (see Fig. 7c). We choose q to be 10 pixels away from s along B's contour, excluding the non-shared boundary $\widehat{ps}$. Then, we connect A and B along the outer boundary $\widehat{tp}$ (see Fig. 7d).

By this co-completion strategy together with the upcoming inflation and stitching steps, we can reconstruct nontrivial double-sided structures, see Figs. 5d and 7d. Note that [16] proposed a method to infer hidden contours from cusps and T-junctions in visible contours, but their method only deals with small suggestive contours and cannot work with the folded structures (our case), which have not been explored in any previous work.

Complete the texture. Lastly, we employ [34] to generate the textures on the completed portions by using the textures sampled from the remainder of the region. These texture samples are chosen to ensure the synthesized results to have minimum variations of luminance, pattern, and structure. Moreover, we fill and inpaint the holes left over on the background image by using [35].

TABLE 1
Icons for Slope and Curvature Markup Cues in our GUI

Icon	Meaning
—🟢▶	Local slope along the out-of-image direction (+Z-axis)
◀🔴▶	Positive mean curvature of surface (convexity)
◀🔵▶	Negative mean curvature of surface (concavity)

5 Inflation

Automatic inference of depth and geometry from a single image is very challenging, due to ambiguities arising in the projection of a 3D scene onto a 2D image. A surface can be planar, convex, concave, or even a mixture of them; even human may fail to interpret its convexity and concavity as in the famous hollow-face dragon illusion. Hence, we take a user-driven approach, aiming at using very little user inputs to quickly reconstruct a plausible high-relief geometry.

5.1 Markup Cues and User Interface

Markup Cues. Before the inflation step, we only have a 2.5D model of the scene, where each (completed) region is planar, i.e., parallel to the image space. To construct 3D information for each of these regions, we consider two very simple user-markup cues as hints for guiding the inflation:

(i) a *slope cue* constrains the local slope on a given region at the marked location, see Table 1(top): the green icon in the GUI and Fig. 8(left). Similar to the out-of-plane tilt handles in [23], our slope cues allow the user to slant the object to pop it out of the image plane. Mathematically, the slope cue has a *position* (the middle dot in the icon), where the cue takes effect, a *direction* (the icon arrow) in 2D, towards which the surface increases in +Z (out-of-image direction), and *length*, which corresponds to the slope magnitude.

(ii) a *curvature cue* constrains the local mean curvature on a given region at the marked position, allowing us to manipulate the local shape profile. We offer two forms of curvature cues: positive and negative, which indicate convexity and concavity, respectively, see Table 1 (middle & bottom) for the related icons. Mathematically, this cue consists of a *position* (the middle dot in the icon), where the cue takes effect, and *length*, which corresponds the amount of curvature. Note that we may also flexibly mark slope and curvature cues on different parts of the same region, see Fig. 8(left and right), for describing more complex shape profiles.

Interaction. We offer three interactions for manipulating the cues, see Fig. 10 (from left to right). First, we may drag the center dot of a cue to relocate it. Second, we may drag the cue's arrowhead to adjust its magnitude (slope or curvature, depending on which cue) and/or direction; note that dragging the arrowhead to the center dot nullifies the effect of the cue. Lastly, dragging the curvature cue's arrowhead across the center dot to the opposite side swaps between positive and negative curvature. Compared to previous works, our method neither requires painting a dense gradient map [18] nor careful and dexterous inputs [22].

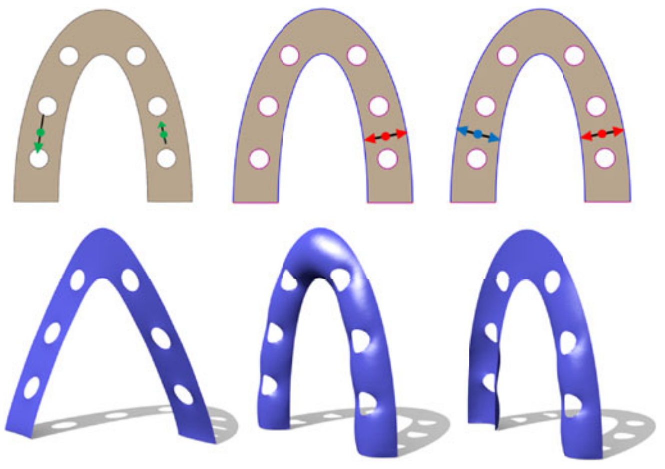


Fig. 8. Top: input images marked up with only slope cues (left) and only curvature cues (middle & right). Bottom: effect of slope cues on boundary vertices (left) and of curvature cues on interior vertices (middle & right) after the inflation. Note that we apply the Dirichlet and Neumann boundary conditions along the blue and purple (bottom of middle&right horseshoes in top row) boundaries, respectively.

User Interface. After the completion step, our interface (GUI) presents two sub-windows, see Fig. 9 (left): the left sub-window shows the completed regions with transparency in flat-shaded rendering style, while the right sub-window shows the inflated high-relief model, which is an interactive feedback to the user. On the left sub-window, the user can select a region and markup the slope and curvature cues. The right sub-window then updates the inflated results, when we add/modify the two cues, see Fig. 9 (middle and right). Note that we sort the regions based on layering order to prevent cluttering of regions caused by intersections. This is a heuristic used for preview only, and Section 6 presents an optimization model to ensure that the high-relief model is free of intersections and gaps.

At the beginning of the inflation step, our interface automatically initializes some slope and curvature cues with certain default values for immediate usage by the user:

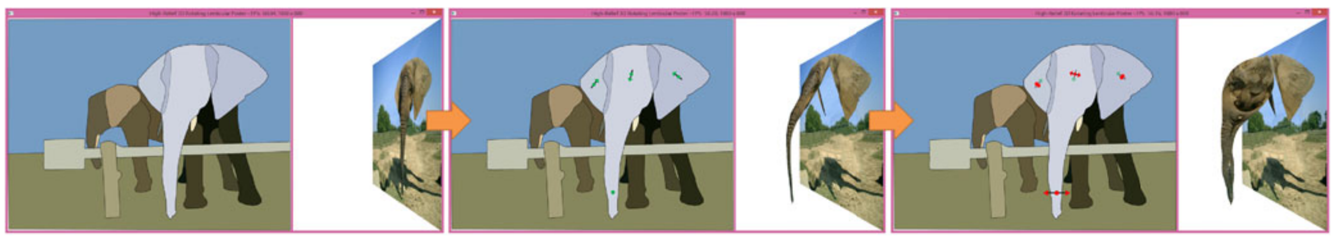


Fig. 9. User workflow in the inflation step. Left: the left sub-window in the GUI shows the completed regions for markup cue manipulation, while the right sub-window previews the inflation result; Middle: inflation result with only slope cues; and Right: result with both slope and curvature cues.

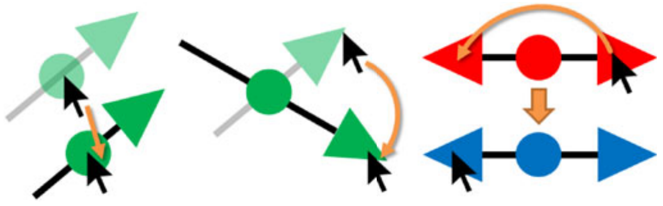


Fig. 10. User interaction with the markup cues. Note: black arrows indicate the mouse cursor. Left: relocate a cue by click & drag its center dot. Middle: modify a cue's magnitude/direction by click & drag its arrowhead. Right: to swap the sign of a curvature cue, click & drag its arrowhead across the center dot all the way to the opposite side.

- At the centroid of each region, we place one slope and one curvature cue; if the centroid lies outside the region, we randomly distribute ten points inside the region and compute the shortest distance from each point to the region boundary. We put the cue at the point with the largest distance among all.
- For slope cue, its default direction is along the major axis in the principal component analysis, and its default magnitude is flat, i.e., planar.
- For curvature cue, its default direction is perpendicular to that of the corresponding slope cue.

On the left sub-window, the user can interactively manipulate the magnitude and direction of each cue, relocate it, or add more cues to the same region to refine the inflation results. Moreover, the user may specify the boundary condition of a region for the inflation to be Dirichlet (fixed position for boundary vertices) or Neumann (zero gradient at boundary vertices along exterior normal), see Fig. 8. Lastly, the user may also specify whether the given region needs a back side. If so, we create and inflate a mirrored version of the given region, and sew it with the inflated geometry of the given region along the region boundary. By this, the reconstructed geometry could become complete when we rotate it at large angles.

5.2 Optimization

Given the user-specified slope and curvature cues, we formulate an optimization model to solve for the geometry of each region with the following goals: i) compute the z-coordinate of the boundary vertices from the user-specified slope cues, and ii) compute the z-coordinate of the interior vertices of each region from the curvature cues (see Fig. 8).

5.2.1 Compute Z-Coordinate of Boundary Vertices

Given an image region $\phi(x, y)$, its gradient can be expressed as $\nabla\phi(x, y) = \vec{\Phi}$, which is to be reconstructed from the slope cues. The desirable slope field should be smooth over the region Ω and curl-free, so that the surface is integrable. Hence, we reconstruct $\vec{\Phi}$ by minimizing the following energy:

$$\min_{\vec{\Phi}} \iint_{\Omega} \left|\nabla\vec{\Phi}\right|^2 + \left|\nabla \times \vec{\Phi}\right|^2 dx\, dy$$

subject to $\vec{\Phi}(x_i, y_i) = \vec{s_i}$, where $\{(x_i, y_i, \vec{s_i})\}$ denotes the set of slope cues in $\phi(x, y)$ and $\times$ denotes cross product. The first term represents the Laplacian, while the second term represents the curl. Obtaining ϕ from $\vec{\Phi}$ is straightforward through integration. We use the obtained slope field $\vec{\Phi}$ to compute the Z-coordinates of the boundary vertices by fixing one of them as 0. These boundary coordinates will be used in the next step to set the Dirichlet boundary conditions while computing the interior vertices.

5.2.2 Compute Z-Coordinate of Interior Vertices

Given a set of regions $\{\phi_i\}$ that form a double-sided structure, we first compute $f^* = \oplus_{i=1}^{m}\phi_i$, where $\oplus$ is an operation that combines the associated regions into a single 2-manifold surface by connecting the shared boundaries (from Case (iii) in Section 4). This allows the inflated double-sided surface to be continuous and smooth across the shared boundary. In case of single-sided structures, we simply set $f^* = \phi_i$.

The mean curvature normal of a surface is $2\kappa\vec{n} = \Delta f$, where Δ is the Laplace-Beltrami operator and f is a 3D surface. The discrete approximation using cotangent weights [36] at each vertex v_i is:

$$\kappa\vec{n} = \frac{1}{4A}\sum_{v_j \in N(v_i)} (\cot\alpha_{ij} + \cot\beta_{ij})(v_i - v_j), \tag{1}$$

where A is the sum of the 1-ring triangle areas around v_i, $N(v_i)$ is the set of 1-ring neighboring vertices of v_i, and α_{ij} and β_{ij} are the angles opposite to edge (v_i, v_j) on either side in the original mesh, see [36].

Similar to [5], we only inflate the surface along z, so we only consider the z-component of the mean curvature normal $\kappa\vec{n}$, which we denote as $\widehat{\kappa}$. To achieve a C^1 continuous surface, particularly across the shared boundary for double-sided structures, we utilize biharmonic equation $\Delta^2 f = \nabla^4 f = 0$. The order of this PDE can be reduced by using two second-order coupled PDEs, i.e., $\Delta f = 2\kappa\vec{n}$ and $\Delta\widehat{\kappa} = 0$. Notably, a similar reduction for the biharmonic equation has been explored in [6], [37]. Our approach is similar to [37], except we employ it to only solve for the z-coordinates of the vertices. There are two advantages: first, solving for only z-coordinates reduces the computation time, and second, this preserves the original 2D projection of the surface. Allowing the vertices to move arbitrarily in 3D space may result in inner vertices that go cross the region boundary if the curvature is large. Nealen et al. [6] used a different strategy to reduce computation time; they replaced the geometry dependent Laplace-Beltrami operator with the uniform Laplacian operator and included additional constraints to preserve the edge lengths. This eliminated the need to compute weights for the discrete Laplacian in each iteration when the geometry changes. In our case, we simply compute the cotangent weights once from the rest mesh and retain these weights in the inflation step.

Here, we first solve for $\Delta\widehat{\kappa} = 0$ by discretizing the equation at each vertex v_i as

$$\sum_{\widehat{\kappa}_j \in N(\widehat{\kappa}_i)} (\cot\alpha_{ij} + \cot\beta_{ij})(\widehat{\kappa}_i - \widehat{\kappa}_j) = 0$$

and the region boundary $\partial\Omega$ subject to the Neumann boundary conditions:

$$\nabla\widehat{\kappa}(\mathbf{x}) \cdot \mathbf{n} = 0 \qquad \forall\mathbf{x} \in \partial\Omega,$$

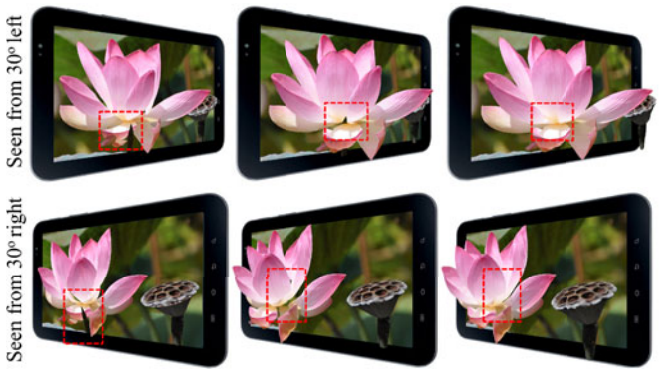


Fig. 11. Stitching. Left: intersection/gaps between regions in the high-relief model before stitching. Middle: stitching result without the second term in the objective function can still contain gaps (highlighted in red). Right: stitching result using the proposed equation.

where $\mathbf{n}$ is the exterior normal to the boundary. The target curvature field $\widehat{\kappa}$ smoothly interpolates the user-specified curvature cues $\{(x_i, y_i, \widehat{\kappa}_i)\}$ over Ω. Next, we recover the inflated 3D surface f by solving the Poisson equation $\Delta f = 2\kappa\vec{n}$ subject to Dirichlet boundary conditions B_D or Neumann boundary conditions B_N, where $B_D \cup B_N = \partial\Omega$:

$$\begin{aligned} f(\mathbf{x}) &= f^* \qquad \forall \mathbf{x} \in B_D \quad \text{and} \\ \nabla f(\mathbf{x}) \cdot \mathbf{n} &= 0 \qquad \forall \mathbf{x} \in B_N. \end{aligned}$$

6 Stitching

After inflating individual regions, we need to arrange the regions along the image normal direction before forming a high-relief model. If this is done inappropriately, we could see undesirable artifacts such as gaps and intersections, and the problem could be exaggerated when different parts of the model overlap (see Fig. 11 (left column)).

To resolve this issue, we optimize the geometry of each region by minimizing the following energy function with respect to hard constraints derived from the relative depth information estimated during the layering step:

$$\min_{f'} \sum_i \left\{ \int_\Omega |\nabla f_i - \nabla f'_i|^2 + |f'_i - Above(f'_i)|^2 dA \right\},$$

subject to:

$$\begin{aligned} f'_i(v_i) &\le f'_j(v_j) \quad \forall (v_i, v_j) \in \{Front(f'_i) \le Back(f'_j)\}, \\ f'_i(v_i) &= f'_j(v_j) \quad \forall (v_i, v_j) \in Stitch(i, j), \end{aligned}$$

where f_i and f'_i are the original (inflated) and deformed height fields of region i, respectively, v_i and v_j are the vertices (image locations) of region i (f_i) and j (f_j), respectively, $Above(\cdot)$ refers to the neighboring region(s) immediately above the current region, $Front(\cdot)$ and $Back(\cdot)$ are the front and back side of a region, respectively, and $Stitch(i, j)$ is the set of vertices along the non-occluding shared boundary (equal depth from the layering step) of i and j (if any).

$Front(\cdot)$ and $Back(\cdot)$ are mainly used for double-sided structures. For regular single-sided geometries, we set $Front(\cdot) = Back(\cdot)$. The first term in the energy function is a fidelity term for minimizing the deformation on each of the height fields, while the second term is for minimizing the size of gaps between regions (see Fig. 11(right column)). The first hard constraint enforces the original 2.5D layering among the regions (see Fig. 11(middle column)) and the second hard constraint glues the neighboring regions along the non-occluding shared boundary between them (if any). Note that the deformation caused by this method is minimum and preserves the shape profile created by the user as much as possible. Since the stitched mesh may have discontinuities along the non-occluding shared boundary between the stitched regions, we use a local smoothing operation around these contours to reduce these artifacts.

7 Results and Discussion

Implementation. To evaluate our method, we implemented it in C++ and evaluated it on a desktop computer with a 3.4GHz CPU and 4GB memory.

Experiment. We recruited six participants (three female and three male; aged 24 to 32) on a volunteering basis to try out our software. Each of them was first given 10 minutes of learning time to get familiar with the software, e.g., how to define and adjust the cues to produce the 3D reconstructions. After that, they can make use of our system to edit the local shape profile and produce various inflation results with some input photos, e.g., Fig. 9.

TABLE 2
Time taken to Generate the High-relief Models Shown in this Paper, Including User Interaction

Inputs	# Triangles	Segmentation		Layering	Completion	Inflation				Stitching	Total time
		# Regions	Time			# Slope & Curvature Cues	Average Compute Time per Region		User annotation		
							Slope	Curvature			
Bird (Fig. 2)	36846	31	9m 51s	0.029s	2.051s	34	0.0258s	0.2133s	4m 40s	1m 8s	15m 41s
Hockey (Fig. 13, row 1)	52396	53	10m 20s	0.050s	3.682s	53	0.0248s	0.1615s	6m 1s	2m 5s	18m 30s
Knot (Fig. 13, row 2)	37178	4	5m 45s	0.014s	1.007s	3	0.0787s	0.4889s	0m 23s	0m 45.3s	6m 54s
Elephant (Fig. 13, row 3)	32948	18	12m 31s	0.018s	1.391s	21	0.0299s	0.2565s	2m 45s	1m 2s	16m 19s
Ballet (Fig. 13, row 4)	33180	23	5m 12s	0.024s	1.503s	30	0.0507s	0.2213s	3m 48s	0m 40.7s	9m 42s
Seacow (Fig. 13, row 5)	24842	5	2m 33s	0.009s	0.308s	7	0.0612s	0.4264s	0m 45s	0m 35.8s	3m 54s
Flower (Fig. 14, row 1)	36597	26	4m 59s	0.021s	1.659s	26	0.1279s	0.4830s	3m 10s	0m 39.2s	8m 50s
Shoes (Fig. 14, row 2)	10974	8	2m 24s	0.008s	0.325s	10	0.0777s	0.5301s	1m 15s	0m 10.3s	3m 50s
Pinwheel (Fig. 14, row 3)	23983	18	6m 17s	0.014s	1.473s	18	0.1403s	0.4465s	2m 53s	0m 10.1s	9m 22s

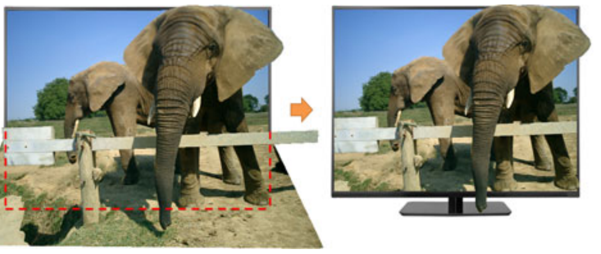


Fig. 12. Post-processing. Left: the high-relief model after inflation and stitching (before cropping). Right: after cropping the ground plane by the extended red box, the result could resemble a lenticular poster.

Overall, they took around 3-18 minutes to reconstruct a high-relief model, depending on the amount of user interactions and the image complexity. The average time taken in the whole process, including the user interactions, is shown in Table 2.

From the experiment, we observe that the interactive feedback mechanism is very important in both the learning process and the editing process. The participants could easily find out how the arrow direction and length affect the 3D reconstruction by interactively modifying the cues and observing the changes in the reconstruction results. Therefore, all the users can achieve a desired local shape profile in around two to three iterations (on average), thanks to the interactive feedback mechanism.

Post-processing. To make the high-relief result resemble a lenticular poster, we further extend the bottom side of the image to form a rectangular region (see the red box with dash lines in Fig. 12(left)), and apply this box to crop the ground plane (see Fig. 12(right)). In addition, we could present the resulting high-relief model optionally with a TV or smart phone frame to enhance the 3D perception.

Results. Fig. 13 shows our high-relief results generated for various organic objects that are without double-sided structures. Here, the first column shows the input (single) image, the second column shows the user-defined cues, the third column shows the reconstructed high-relief models,

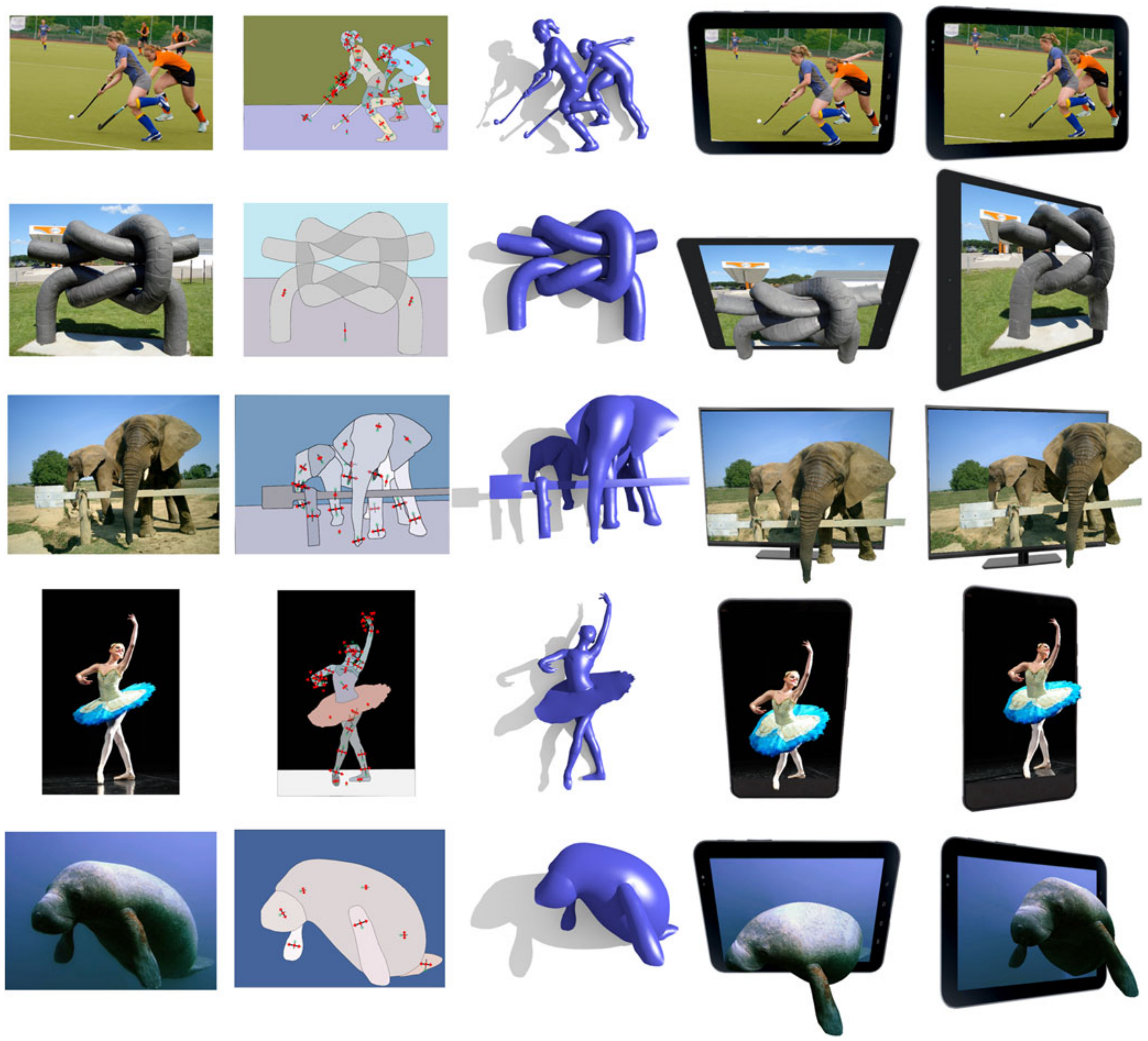


Fig. 13. Results for single-sided structures: Hockey, Knot, Elephant, Ballet, and Seacow (rotated 12 to 45 degree from the original viewing direction).

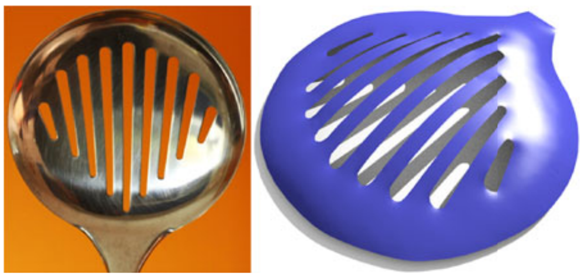


Fig. 14. Our method can handle objects with holes (within a segmented image region). Left: input image. Right: inflated result.

while the last two columns show novel views of the models with large rotation from the original view. Fig. 14 presents another example, demonstrating that our method can also handle foreground objects with holes.

Fig. 15 shows our high-relief results generated for objects with double-sided structures. These results are completed by our method in Section 4 and later inflated and stitched by our methods in Sections 5 and 6. Since we can only show static images in the paper, we present animated versions of these high-relief results in the supplementary video which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TVCG.2016.2574705).

Comparison. Next, we compare our results with two closely-related recent works: [4] and [5]. In particular, we compare the visual plausibility of the results, with large rotations from the original viewing direction (please refer to the supplementary video, available online, for animated versions of the comparison).

Zeng et al. [4] presented a method to generate stereoscopic effects from a single segmented image. However, their method can only allow small rotation angles (7 degree) since their 3D mesh is flat. Our results still have plausible 3D appearance even with large rotations (see Fig. 16).

The method proposed by Sýkora et al.[5] was mainly designed for creating a proxy mesh of the object in the input image and then enhancing it with global illumination. Hence, while the resulting bas-relief meshes appear visually plausible in the original view from the front, it is revealed to be flat after rotations (see Fig. 17). As a comparison, our high-relief mesh results can achieve a pop-out visual effect in the side views.

Limitations. We discuss the limitations of this work as related to different steps in the method:

i) *Segmentation:* some images can be too complex for proper segmentation, e.g., thin and fine structures (see Figs. 18b and 18c); hence, we cannot form meaningful regions and compute layering.
ii) *Local layering:* our method cannot automatically recover the layering order in an intertwined structure since it involves more than one single layering order between certain neighboring image regions (see red circles in Fig. 18d); users can manually divide the edge and correct the layering.
iii) *Completion:* our image completion is a best-effort step, which tries to estimate the occluded part of a region based on what is visible. Since it does not involve semantics, it cannot connect hidden regions arbitrarily.
iv) *Inflation:* our method is mainly designed for reconstructing smooth and organic shapes, where the inflation model assumes smooth surfaces (see Fig. 18a), so we need extra constraints to properly handle objects with sharp edges. Moreover, our current method supports only a parabolic profile, perfect spherical shapes cannot be generated.
v) *Stitching:* our method generally works well in stitching, except for ill-completed meshes, where we lack information to correctly compute the stitching.

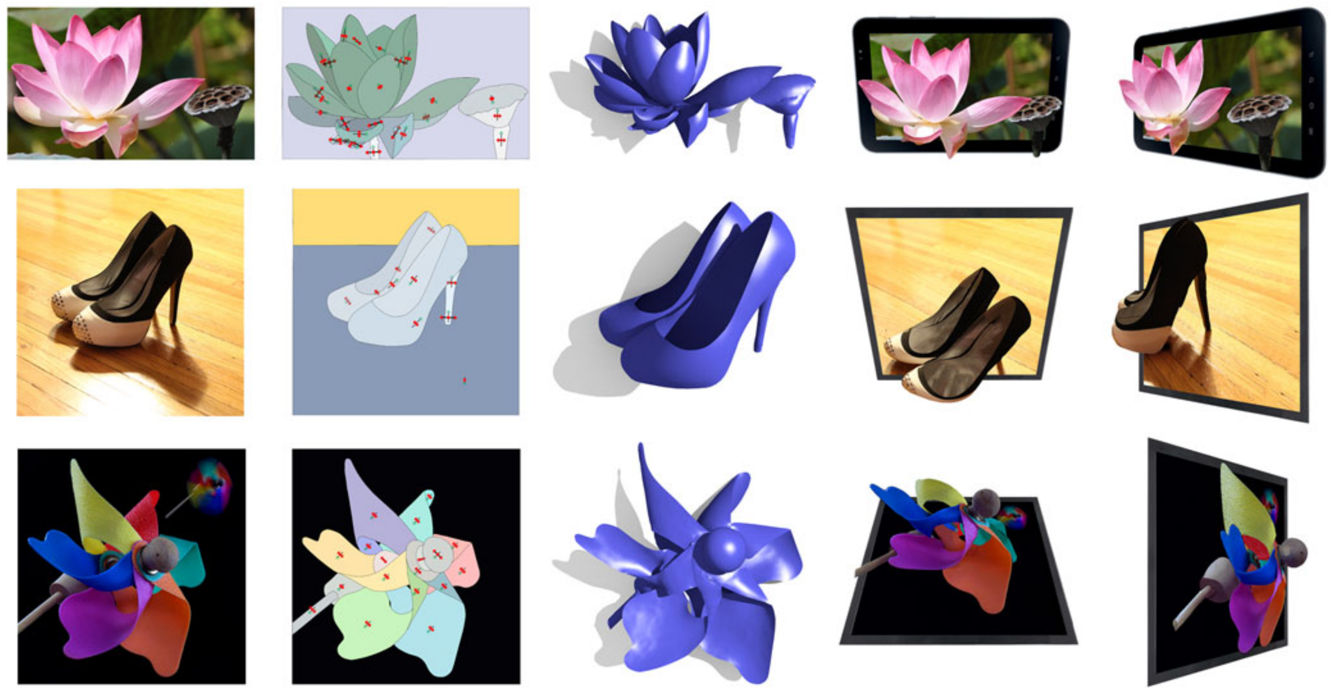


Fig. 15. Results for double-sided structures. Flower, Shoe, and Pinwheel (rotated 30 to 45 degree from the original viewing direction).

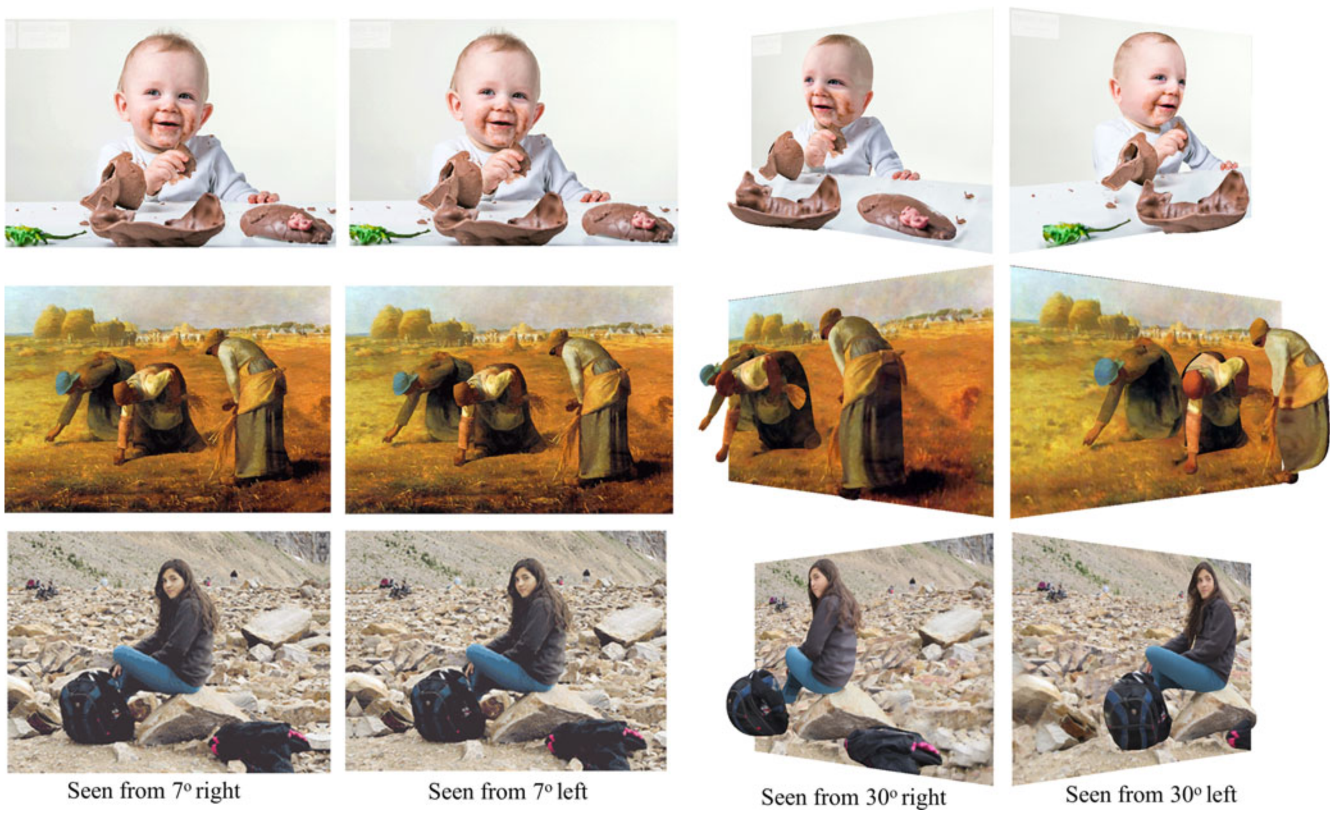


Fig. 16. Comparison with [4]. Columns 1 & 2: results from [4] with ∼7 degree rotations downloaded from their website, and Columns 3 & 4: our results with much larger rotations. Please refer to the supplementary video, available online, for animated version of this comparison.

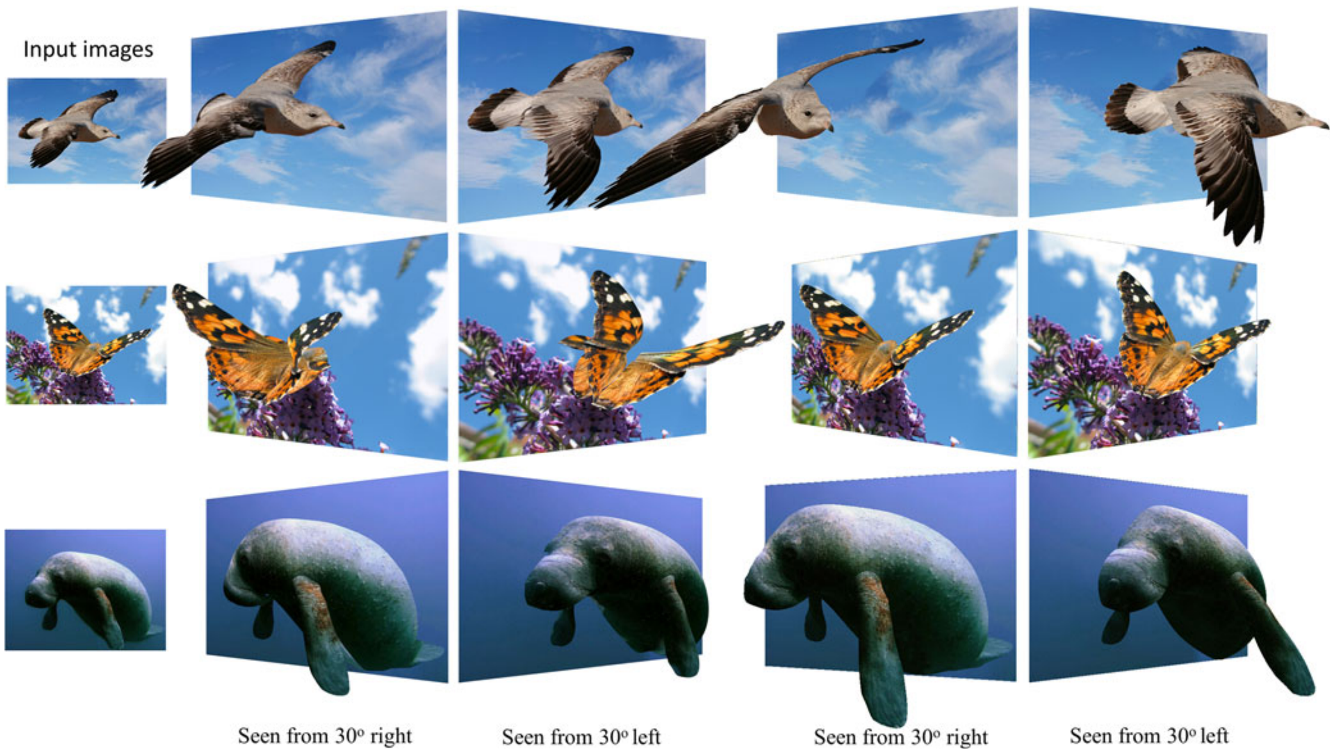


Fig. 17. Comparison with [5]. Columns 1 & 2: results kindly generated by Daniel Sýkora using his method in [5], and Columns 3 & 4: our results with more plausible geometry under same amount of 3D rotations from the original view. Please also refer to the supplementary video, available online.

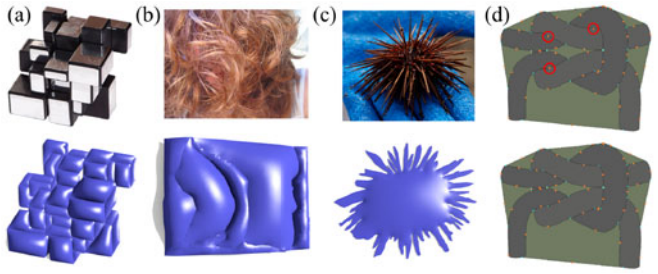


Fig. 18. Limitations. (a) polyhedral shape with sharp edges, (b & c) objects with very fine details are hard to be segmented, (d) intertwined regions cannot be consistently estimated by our local layering step.

8 Conclusion

This paper presents a novel approach to reconstruct high-relief 3D models from a single input image, aiming at achieving large view rotation on the reconstructed results. We particularly consider common organic objects with non-trivial shape profile and the reconstruction of objects composed of double-sided structures. These are novel elements that we focus on in this research work.

In summary, this work has the following technical contributions. First, we develop a completion method that considers three common cases of occlusion found in natural images; particularly, we are able to complete the geometry of double-sided structures in different cases. Second, we devise an interactive inflation model; taking little amount of user inputs (slope and curvature cues), we can perform a constrained optimization to quickly construct an inflated shape profile with large rotation for interactive editing with preview. We also develop a stitching method that respects the layering and double-sided information to aid the high-relief reconstruction. Lastly, we show the superiority of our results by comparing it with two very recent work and present lenticular poster renderings of our results for a wide variety of common organic objects.

Future Work. In future, we plan to investigate methods to automate the estimation of slope and curvature cues from the image contents, e.g., by analyzing the local shading. Second, we would study how sharp features may integrate into our computational framework, so that it can simultaneously handle both organic and polyhedron shapes.

Acknowledgments

We would like to thank the reviewers for the many constructive comments that help improve the paper. We are grateful to Daniel Sýkora and Olga Sorkine-Hornung for their help in generating the comparison results, and the authors of [4] for releasing their dataset and results online. This work was supported in part by the Ministry of Science and Technology (contracts MOST-104-2221-E-006-044-MY3 and MOST-103-2221-E-006-106-MY3), Taiwan, and the Chinese University of Hong Kong strategic recruitment fund and direct grant (4055061).

References

[1] Wikipedia, Relief. [Online]. Available: http://en.wikipedia.org/wiki/Relief#High_relief, 2013. Accessed on: Mar. 17, 2015.

[2] P. Cignoni, C. Montani, and R. Scopigno, “Computer-assisted generation of bas- and high-reliefs,” *J. Graph. Tools*, vol. 2, no. 3, pp. 15–28, 1997.

[3] D. Hoiem, A. A. Efros, and M. Hebert, “Automatic photo pop-up,” *ACM Trans. Graph.*, vol. 24, no. 3, pp. 577–584, 2005.

[4] Q. Zeng, W. Chen, H. Wang, C. Tu, D. Cohen-Or, D. Lischinski, and B. Chen, “Hallucinating stereoscopy from a single image,” *Comput. Grap. Forum*, vol. 34, no. 2, pp. 1–12, 2015.

[5] D. Sýkora, et al., “Ink-and-ray: Bas-relief meshes for adding global illumination effects to hand-drawn characters,” *ACM Trans. Graph.*, vol. 33, no. 2, pp. 16:1–15, 2014.

[6] A. Nealen, T. Igarashi, O. Sorkine, and M. Alexa, “FiberMesh: Designing freeform surfaces with 3D curves,” *ACM Trans. Graph.*, vol. 26, no. 3, pp. 41:1–8, 2007.

[7] P. Joshi and N. A. Carr, “Repoussé: Automatic inflation of 2D artwork,” in *Proc. 15th Eurographics Conf. Sketch-Based Interfaces Model.*, 2008, pp. 49–55.

[8] F. Yan, M. Gong, D. Cohen-Or, O. Deussen, and B. Chen, “Flower reconstruction from a single photo,” *Comput. Graph. Forum*, vol. 33, no. 2, pp. 439–447, 2014.

[9] A. Saxena, M. Sun, and A. Ng, “Make3D: Learning 3D scene structure from a single still image,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 5, pp. 824–840, 2009.

[10] T. Chen, Z. Zhu, A. Shamir, S.-M. Hu, and D. Cohen-Or, “3-sweep: extracting editable objects from a single photo,” *ACM Trans. Graph.*, vol. 32, no. 6, pp. 195:1–10, 2013.

[11] N. Kholgade, T. Simon, A. Efros, and Y. Sheikh, “3D object manipulation in a single photograph using stock 3D models,” *ACM Trans. Graph.*, vol. 33, no. 4, pp. 127:1–12, 2014.

[12] D. Hoiem, A. Efros, and M. Hebert, “Geometric context from a single image,” in *Proc. 10th IEEE Int. Conf. Comput. Vis.*, 2005, vol. 1, pp. 654–661.

[13] D. Hoiem, A. Efros, and M. Hebert, “Recovering surface layout from an image,” *Intl. J. Comput. Vis.*, vol. 75, no. 1, pp. 151–172, 2007.

[14] J.-D. Durou, M. Falcone, and M. Sagona, “Numerical methods for shape-from-shading: A new survey with benchmarks,” *Comput. Vis. Image Underst.*, vol. 109, no. 1, pp. 22–43, 2008.

[15] T. Igarashi, S. Matsuoka, and H. Tanaka, “Teddy: A sketching interface for 3D freeform design,” in *Proc. 26th Annu. Conf. Comput. Graph. Interact. Tech.*, 1999, pp. 409–416.

[16] O. A. Karpenko and J. F. Hughes, “SmoothSketch: 3D free-form shapes from complex sketches,” *ACM Trans. Graph.*, vol. 25, no. 3, pp. 589–598, 2006.

[17] T. Ijiri, S. Owada, and T. Igarashi, “Seamless integration of initial sketching and subsequent detail editing in flower modeling,” *Comput. Graph. Forum*, vol. 25, no. 3, pp. 617–624, 2006.

[18] C. Van Overveld, “Painting gradients: Free-form surface design using shading patterns.” in *Proc. Graph. Interface*, 1996, pp. 151–158.

[19] B. M. Oh, M. Chen, J. Dorsey, and F. Durand, “Image-based modeling and photo editing,” in *Proc. 28th Annu. Conf. Comput. Graph. Interact. Tech.*, 2001, pp. 433–442.

[20] L. Zhang, G. Dugas-Phocion, J.-S. Samson, and S. M. Seitz, “Single-view modelling of free-form scenes,” *J. Vis. Comput. Animat.*, vol. 13, no. 4, pp. 225–235, 2002.

[21] F. Cordier and H. Seo, “Free-form sketching of self-occluding objects,” *IEEE Comput. Graph. Appl.*, vol. 27, no. 1, pp. 50–59, 2007.

[22] T.-P. Wu, C.-K. Tang, M. S. Brown, and H.-Y. Shum, “ShapePalettes: Interactive normal transfer via sketching,” *ACM Trans. Graph.*, vol. 26, no. 3, 2007.

[23] Y. Gingold, T. Igarashi, and D. Zorin, “Structured annotations for 2D-to-3D modeling,” *ACM Trans. Graph.*, vol. 28, no. 5, pp. 148:1–9, 2009.

[24] M. Prasad, A. Zisserman, and A. W. Fitzgibbon, “Fast and controllable 3D modelling from silhouettes,” in *Proc. Annu. Conf. Eur. Assoc. Comput. Graph.*, Sep. 2005, pp. 9–12.

[25] M. Prasad and A. Fitzgibbon, “Single view reconstruction of curved surfaces,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, vol. 2, pp. 1345–1354, 2006.

[26] Y. Zheng, X. Chen, M.-M. Cheng, K. Zhou, S.-M. Hu, and N. J. Mitra, “Interactive images: Cuboid proxies for smart image manipulation,” *ACM Trans. Graph.*, vol. 31, no. 4, pp. 99:1–11, 2012.

[27] Y. Horry, K.-I. Anjyo, and K. Arai, “Tour into the picture: Using a spidery mesh interface to make animation from a single image,” in *Proc. 24th Annu. Conf. Comput. Graph. Interact. Tech.*, 1997, pp. 225–232.

[28] A. Rivers, T. Igarashi, and F. Durand, “2.5D cartoon models,” *ACM Trans. Graph.*, vol. 29, pp. 59:1–7, 2010.

[29] C.-K. Yeh, P. Jayaraman, X. Liu, C.-W. Fu, and T.-Y. Lee, “2.5D cartoon hair modeling and manipulation,” *IEEE Trans. Vis. Comput. Graph.*, vol. 21, no. 3, pp. 304–314, 2015.

[30] L. Zhang, H. Huang, and H. Fu, "EXCOL: An EXtract-and-COmplete Layering approach to cartoon animation reusing," *IEEE Trans. Vis. Comput. Graph.*, vol. 18, no. 7, pp. 1156–1169, 2012.

[31] X. Liu, X. Mao, X. Yang, L. Zhang, and T.-T. Wong, "Stereoscopizing cel animations," *ACM Trans. Graph.*, vol. 32, no. 6, pp. 223:1–10, 2013.

[32] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 9, pp. 1124–1137, 2004.

[33] S. Osher and R. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*. Berlin, Germany: Springer Science & Business Media, 2003.

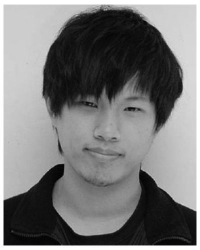
[34] A. Criminisi, P. Perez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Trans. Image Process.*, vol. 13, no. 9, pp. 1200–1212, 2004.

[35] M. Daisy, D. Tschumperlé, and O. Lézoray, "A fast spatial patch blending algorithm for artefact reduction in pattern-based image inpainting," in *Proc. SIGGRAPH Asia Tech. Briefs*, 2013, pp. 8:1–4.

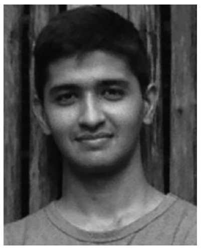
[36] M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr, "Implicit fairing of irregular meshes using diffusion and curvature flow," in *Proc. 26th Annu. Conf. Comput. Graph. Interact. Tech.*, 1999, pp. 317–324.

[37] R. Schneider and L. Kobbelt, "Geometric fairing of irregular meshes for free-form surface design," *Comput. Aided Geom. Des.*, vol. 18, no. 4, pp. 359–379, 2001.

Chih-Kuo Yeh received the BS degree from the Department of Information Engineering and Computer Science, Feng Chia University, in 2005, the MS degree from the Institute of Bioinformatics, National Chiao Tung University, in 2007 and the PhD degree from the Department of Computer Science and Information Engineering, National Cheng-Kung University, Taiwan, in 2015. He is currently a postdoctoral researcher in National Cheng-Kung University. His research interests include scientific visualization, computer animation, and computer graphics.



Shi-Yang Huang received the BS degree from the Department of Information Engineering and Computer Science, National Taipei University of Technology, in 2013 and the MS degree from the Department of Computer Science and Information Engineering, National Cheng-Kung University, Taiwan, in 2015. His research interests include visualization and computer graphics.



Pradeep Kumar Jayaraman received the BTech degree in information technology from Anna University, India in 2009. He is currently working toward the PhD degree from the School of Computer Science and Engineering, Nanyang Technological University, Singapore. His research interests lie at the intersection of computer vision, computer graphics, and machine learning.

Chi-Wing Fu received the BSc and MPhil degrees both in computer science and engineering from The Chinese University of Hong Kong, and the PhD in computer science from Indiana University, Bloomington, USA. He is currently an associate professor in the Department of Computer Science and Engineering, the Chinese University of Hong Kong. His research focuses on interactive methods in the areas of computer graphics, geometric design, visualization, and human-computer interaction. He has served as an associate editor of Computer Graphics Forum. He is a member of the IEEE.

Tong-Yee Lee received the PhD degree in computer engineering from Washington State University, Pullman, in May 1995. He is currently a chair professor in the Department of Computer Science and Information Engineering, National Cheng-Kung University, Tainan, Taiwan, ROC. He leads the Computer Graphics Group, Visual System Laboratory, National Cheng-Kung University (http://graphics.csie.ncku.edu.tw/). His current research interests include computer graphics, nonphotorealistic rendering, medical visualization, virtual reality, and media resizing. He is a senior member of the IEEE Computer Society and a member of the ACM.