Pacific Graphics 2014
J. Keyser, Y. J. Kim, and P. Wonka
(Guest Editors)

Volume 33 (2014), Number 7

Interactive Image-Guided Modeling of Extruded Shapes

Yan-Pei Cao[1] Tao Ju[2] Zhao Fu[1] Shi-Min Hu[1]

[1]Tsinghua National Laboratory for Information Science and Technology
and Department of Computer Science and Technology, Tsinghua University
[2]Washington University in St. Louis

Abstract

A recent trend in interactive modeling of 3D shapes from a single image is designing minimal interfaces, and accompanying algorithms, for modeling a specific class of objects. Expanding upon the range of shapes that existing minimal interfaces can model, we present an interactive image-guided tool for modeling shapes made up of extruded parts. An extruded part is represented by extruding a closed planar curve, called base*, in the direction orthogonal to the base. To model each extruded part, the user only needs to sketch the projected base shape in the image. The main technical contribution is a novel optimization-based approach for recovering the 3D normal of the base of an extruded object by exploring both geometric regularity of the sketched curve and image contents. We developed a convenient interface for modeling multi-part shapes and a method for optimizing the relative placement of the parts. Our tool is validated using synthetic data and tested on real-world images.*

Categories and Subject Descriptors (according to ACM CCS): I.3.m [Computer Graphics]: Miscellaneous—image-based modeling

1. Introduction

3D modeling plays a fundamental role in computer graphics. The field has been re-invigorated by recent applications, particularly 3D printing, which puts modeling in the hands of *novice* users. Unlike experienced 3D modelers, novices do not have sophisticated knowledge of geometry nor experience in using complicated modeling software. As a result, the modeling process has to be made intuitive to understand and easy to use.

With the wide availability of digital cameras and public image repositories, a common scenario of novice modeling is to re-create a 3D object in a single 2D image either taken by the user herself or found online. An important advantage of being guided by an image, compared to using a classical modeling tool such as Maya, is that the user does not have to come up with the design from scratch.

While humans have innate ability of perceiving 3D shapes from 2D pictures, computers still have a long way to catch up. As a result, many efforts seek to marry the cognitive power of human users with computational algorithms. A recent trend in this direction is designing *minimal* interfaces that are specialized for modeling a class of objects. These objects include cuboids [ZCC*12], tubular shapes [CZS*13], symmetric architecture [JTC09], and symmetric shapes with planar facets [XLT12]. These methods require only a small amount of effort from the user but are capable of creating accurate models of the respective class.

In this paper, we present a new interactive image-guided tool that expands the capability of existing minimal interfaces. Our tool is specially designed for modeling objects made up of *extruded* parts, each represented by a planar face (called the *base*) that is extruded in the direction orthogonal to that face. The rationale of choosing extruded shapes to model is two-fold. First, they are commonly found in man-made objects, such as furniture, architecture and CAD objects. Note that the base of an extruded object can have an arbitrary 2D shape with possible curved sides. Second, these shapes have a simple definition; they can be completely represented by the shape and location of the base plus the amount of extrusion (i.e., thickness). Compared to general free-form objects, the simplicity of extruded parts makes it possible to design minimal user interfaces as well as more robust and efficient reconstruction algorithms.

The modeling process in our tool is illustrated in Figure 1. The input is an image with calibrated camera parameters. To model an extruded object, the user only needs to trace the outline of the base and indicate the extrusion direction.

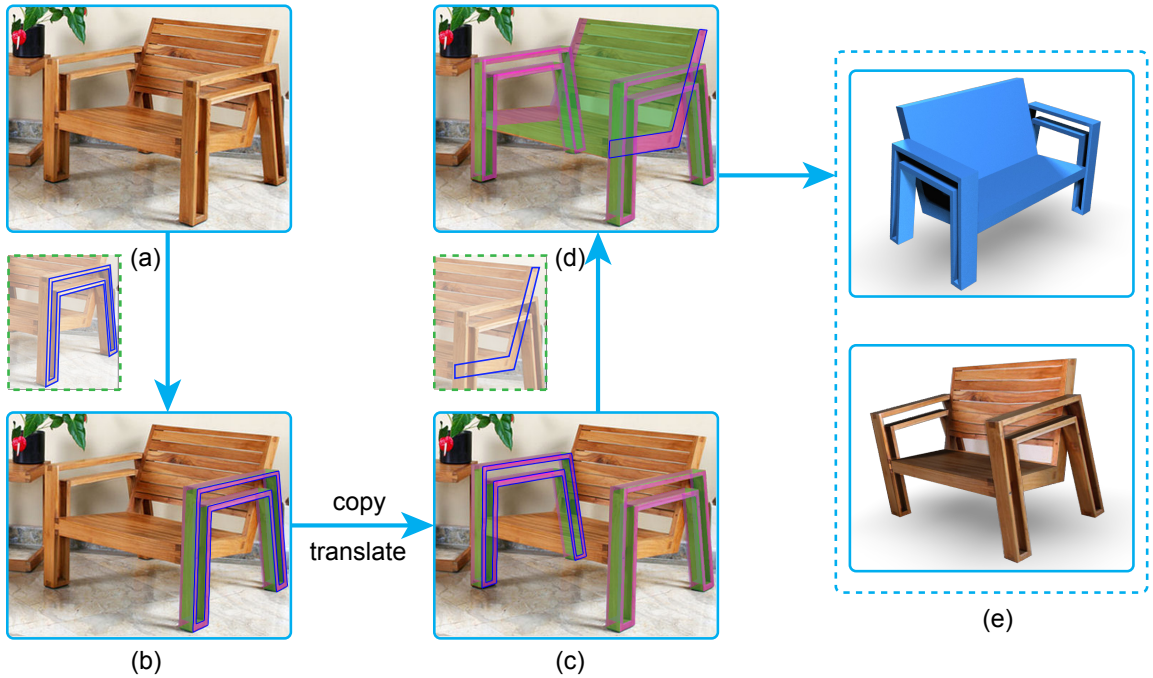

Figure 1: *Modeling pipeline. Starting from an image (a), the user traces the base of an extruded part (shown in insert), from which the system reconstructs the 3D shape (b). The user creates more parts by either copying and transforming (in 3D) existing parts (c) or modeling the new part individually (d). The system optimizes the placement of the parts and visualizes the resulting model with texture (e).*

The system automatically constructs the 3D model up to an unknown depth from the view plane (see (a)→(b)). For a multiple-part object, each part can be either modeled individually (see (c)→(d)) or copied and transformed from an existing part (see (b)→(c)). The system automatically determines the depth of each part. The final model can be viewed and textured (see (e)).

The technical contribution of the work is two-fold. First, we propose an optimization-based method for reconstructing an extruded shape from its projected base shape and extrusion direction. Our optimization formulation combines regularity constraints on the base shape, which are borrowed from and extend upon the literature on wireframe reconstruction, with constraints from the image contents. Second, we propose a method for optimizing 3D placements of multiple parts, which combines regularity constraints between parts, image content, and optional user inputs (e.g., a contact edge or point between two parts).

The paper is organized as follows. After reviewing previous work in interactive image-based modeling (Section 2), we detail the camera calibration (Section 3), the modeling of a single extruded part (Section 4), and modeling a multi-part object (Section 5). Results are then presented and discussed (Section 6). Please also check out the complimentary video that demonstrates the tool.

2. Related Work

There is a large volume of work on interactive 3D modeling and image-based modeling. Given the scope of our paper, we explore the overlap space between the two areas, namely interactive modeling from a single image. Our review will focus on the type of the user inputs required by these methods and the range of their output models.

At one end of the spectrum, several tools require 3-dimensional inputs from the users. Tsang et al. [TBSR04] developed a 3D curve sketching tool guided by an image, and Oh et al. [OCDD01] let the user paint depth map directly into the image to help recover the scene. However, these inputs are not only time-consuming to supply but also challenging for novice users.

In a less laborious setting, the user can draw the 2D wireframe of the object in the image, from which a 3D shape is reconstructed by the system. Interpretation of 2D wireframes is a classical and challenging problem in computer vision

[Sug86]. Successful methods exploit regularity assumptions that are specific to a class of models, such as CAD models [VM00, LS07, WCLT09], architecture [CKX*08], symmetric objects [OUP*11], and conceptual designs [XCS*14]. The user can also supply the regularity terms (e.g., orthogonality and parallelism) or additional cues (e.g., spatial grids) through the user interface [LSMI10, ZLL12].

Drawing a complete wireframe can be a tedious and error-prone task. To simplify the input, some methods allow users to supply only a rough sketch of the desired shape and retrieve similar shapes from a database [ERB*12, XCF*13]. The recent method of Xu et al. [XZZ*11] deforms the retrieved model to fit that in the image. Chen et al. [CGZZ13] estimates a person's shape from a single image using a deformable model. While convenient to use, these methods do not in general recover the same object from the underlying image.

Another school of methods, to which our method belongs, designs minimal interfaces (and accompanying algorithms) that are specialized for a certain class of objects. Zheng et al. [ZCC*12] reconstructs cuboid objects in an image given segmented hexagonal boundary provided by the user. Chen et al. [CZS*13] reconstructs tubular shapes from user strokes along the shape and at its base. Jiang et al. [JTC09] developed an interactive system for image-guided architecture modeling, which utilizes the typical frustum-like symmetry of architecture and user strokes that define an initial model based on such symmetry. Xue et al. [XLT12] reconstructs a bilaterally symmetric piece-wise planar object in an image given user markings of symmetric line pairs and depth discontinuity. While the former three methods produce surface geometry representing the object, the last method produces a point cloud that approximates the shape.

The extruded objects considered in this work cannot be easily modelled using existing minimal interfaces. While an extruded object is by definition bilaterally symmetric, it does not in general have the frustum-like symmetry as in architecture, and the surface does not need to be piece-wise linear (see Figure 7). Extruded objects are also non-trivial generalization of those tubular objects with a pre-defined base shape such as a square or a circle. While previous tools [ZCC*12, CZS*13] can obtain the orientation of the base plane from simple user inputs and prior knowledge of the base shape, the orientation task is much more difficult for an arbitrary and unknown base shape, a key problem that will be addressed in this work.

Interactive modeling of extruded geometry has also appeared in architectural design [KW11] and facade modeling [FWZQ13]. While these tools require images or sketches from different views (as in the silhouette modeling interface of [RDI10]), our tool only needs a single image and sketches in that view.

3. Camera model and calibration

For simplicity, we assume a simplified camera model with zero skew and radial distortion. Thus the projection is completely determined by the focal length of the camera and a translation vector on the view plane.

The camera parameters can be obtained automatically using vanishing point methods [HZ03]. However, detecting and using vanishing point can be numerical unstable [WSB05]. For improved robustness, we follow the user-guided approach in [XLT12]. The user helps to solve the camera parameters by supplying two corners in the image, each with a point and three axes directions, and adjusting the projected parallel lines.

4. Modeling an extruded object

An extruded shape has a 2.5D structure that lifts one planar face in the orthogonal direction. As explained in Figure 2, we call the planar face that is oriented towards the viewer (or camera) as the *base* and the other identical, but backward oriented face as the *cap*. The remaining part of the shape is called the *side*, which is made up of a one-parameter family of parallel line segments called *rails*. Note that an extruded shape in our definition is a special case of generalized cylinders with a straight axis and identical cross-sections.

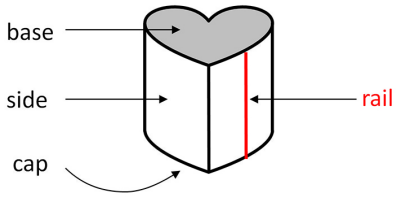

Figure 2: *Notions on an extruded shape.*

Our algorithm assumes the presence of certain regularity in the shape of the base; it needs to have either parts with bilateral symmetry or straight sides that are parallel or orthogonal to each other. We have observed that such assumption usually holds for man-made extruded objects in the real-world (with the notable exception of creative designs). In addition, our algorithm exploits the boundary curve of both the base and cap in the image. Hence we require that the base and the cap must be partly invisible (e.g., not completely occluded or hidden).

To model an extruded object, the user needs to sketch the base and one rail in the image, both are 2D projections of their 3D counterparts. Our algorithm then constructs the 3D geometry of the object up to its distance from the camera origin.

4.1. User interface

To outline the base curve, the user can either draw straight segments or manually trace the image edges. For more accurate tracing, we implemented a snapping feature akin to Photoshop's magnetic lasso. We first apply a fast edge detector [DABP14] to the image, then use the technique in [Che09] to create continuous curves (which we call "creases") from the detected edge pixels. When the user's cursor moves near a crease segment, the segment is automatically selected and connected to the existing sketch. As an example, the base curve in Figure 3 (b) was traced using our snapping tool (see also the accompanying video).

The user can either manually draw a rail or select from those detected by the system (Figure 3 (c)). To detect a rail, we search for straight edges in the creases detected above that are near sharp corners of the base curve. A rail can be selected simply by moving the cursor near it and click to confirm.

4.2. Problem formulation

Given the 2D base curve and a 2D rail, the 3D extruded object can be completely determined by the 3D plane on which the base lies (called the *base plane*). Projecting the 2D base and rail respectively onto the base plane and its normal direction gives the 3D base and the 3D rail.

While any choice of the base plane could give rise to an extruded object, we seek one that maximizes the regularity of the base shape (on the base plane) and conformation to the image. Specifically, we look for a base with strong partial symmetry and whose straight edges, if they exist, are as orthogonal or parallel as possible. Also, the cap of the object should be aligned with strong image edges (the creases). We formulate an optimization problem that seeks the base plane that maximizes the sum of these objectives.

We shall detail each objective below before presenting the optimization formulation at the end. Note that since our objectives are invariant with the translation of the base plane in the viewing direction, the only variable in our optimization reduces to the normal vector of the base plane, which we denote as $\mathbf{n}$. We assume that the plane passes through some fixed point in space, which will be adjusted in the presence of multiple objects (see next section).

Parallelism This objective measures the amount of parallelism among the straight edges of the base curve. We first detect straight segments along the sketched 2D base. For each segment s, and given the normal $\mathbf{n}$ of the base plane, we denote $s_{\mathbf{n}}$ as the 3D line segment on the base plane that projects onto s. This objective measures, for a normal $\mathbf{n}$ and a set of candidate pairs of segments S_p, the total amount of parallelism for each pair of segments $\{s,t\} \in S_p$ (the choice of S_p will be discussed in Section 4.3):

$$f_p(\mathbf{n}, S_p) = \frac{1}{|S_p|} \sum_{\{s,t\} \in S_p} \exp\left(-\frac{(\theta(s_{\mathbf{n}}, t_{\mathbf{n}}))^2}{\sigma_1^2}\right) \omega(s_{\mathbf{n}}, t_{\mathbf{n}}) \quad (1)$$

Here, $|\cdot|$ is the cardinality of a set, θ measures the acute angle between the two line segments in 3D, and ω is a weighting term that gives more weight to segments that are closer and longer. We use the following definition

$$\omega(s_{\mathbf{n}}, t_{\mathbf{n}}) = \exp\left(-\frac{d(s_{\mathbf{n}}, t_{\mathbf{n}})^2}{(\sigma_2 \cdot L)^2}\right) \frac{l(s_{\mathbf{n}}) l(t_{\mathbf{n}})}{L^2},$$

where d measures the distance between the centers of two segments, l gives the length of a segment, and L is the maximum length among $s_{\mathbf{n}}$ for all segments s that appear in S_p. In our implementation we used $\sigma_1 = 0.1, \sigma_2 = 0.3$.

Orthogonality Similar to parallelism, orthogonality is measured for a candidate set of pairs of straight segments on the sketched base curve, noted as S_o. The objective has a similar form,

$$f_o(\mathbf{n}, S_o) = \frac{1}{|S_o|} \sum_{\{s,t\} \in S_o} \exp\left(-\frac{(\pi/2 - \theta(s_{\mathbf{n}}, t_{\mathbf{n}}))^2}{\sigma_1^2}\right) \omega(s_{\mathbf{n}}, t_{\mathbf{n}}), \quad (2)$$

where the weighting function ω is defined in the same way as in the parallelism objective.

Partial symmetry Symmetry has been exploited previously for determining face orientation in wireframe reconstruction [VM02, LLT07]. These methods usually assume global bilateral symmetry. We relax the assumption to handle base shapes with only partial bilateral symmetry. Our partial symmetry measure is similar to that in shape symmetrization [MGP07]. While symmetrization uses the measure to locate the symmetry axes and symmetric parts, we use it to find the base plane that give rises to the most symmetric base shape.

We uniformly sample the sketched base curve into points. For each point p, and given the base plane normal $\mathbf{n}$, we denote $p_{\mathbf{n}}$ as the 3D location on the base plane that projects onto p. Similar to parallelism and orthogonality, we measure symmetry in a set of point pairs S_s. Given two pairs of points $\{p,q\} \in S_s$ and $\{p',q'\} \in S_s$, we compute the *asymmetry* of the 3D point pair $p'_{\mathbf{n}}, q'_{\mathbf{n}}$ with respect to the symmetry plane E of $p_{\mathbf{n}}, q_{\mathbf{n}}$ as follows

$$asym(p_{\mathbf{n}}, q_{\mathbf{n}}, p'_{\mathbf{n}}, q'_{\mathbf{n}}) = \frac{(p'_{\mathbf{n}} - m_E(q'_{\mathbf{n}}))^2}{(p'_{\mathbf{n}} - q'_{\mathbf{n}})^2}$$

where $m_E(\cdot)$ denotes the reflection of a point by the symmetry plane E. Our symmetry term considers all such pairs of point pairs in S_s,

$$f_s(\mathbf{n}, S_s) = \frac{1}{|S_s|^2} \sum_{\{p,q\} \in S_s} \sum_{\{p',q'\} \in S_s} \exp\left(-\frac{(asym(p_{\mathbf{n}}, q_{\mathbf{n}}, p'_{\mathbf{n}}, q'_{\mathbf{n}}) \;\; curv(p_{\mathbf{n}}, q_{\mathbf{n}}))^2}{\sigma_3^2}\right), \quad (3)$$

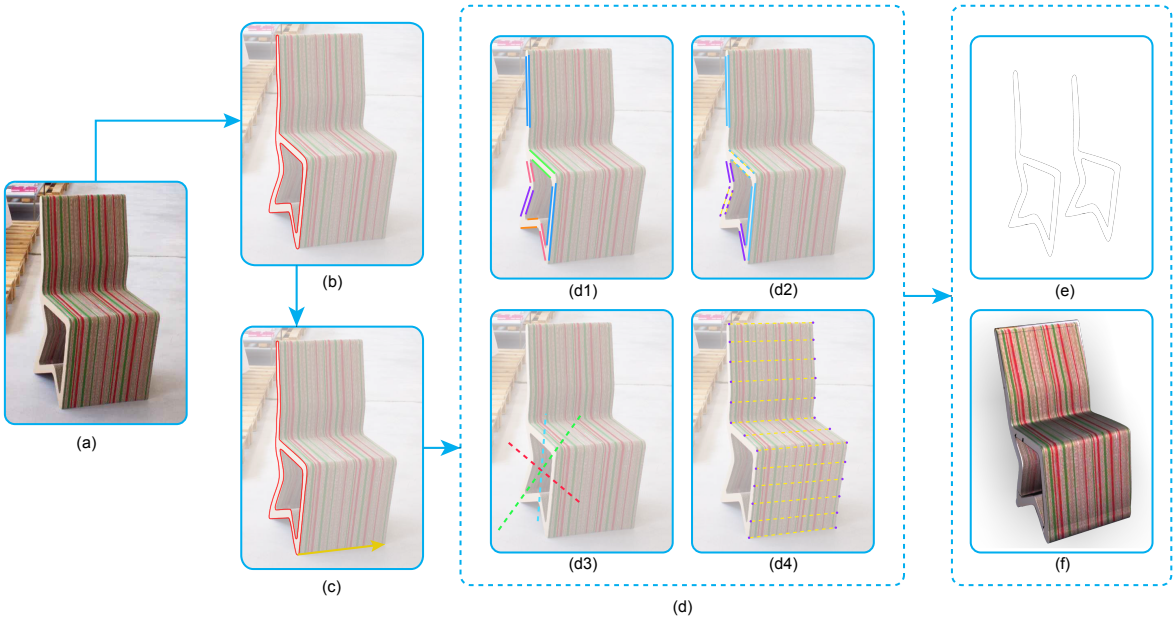

Figure 3: *Modeling an extruded object. Starting from a calibrated image (a), the user sketches the base curve (b) and picks a rail (c). The algorithm first identifies candidate pairs of lines/points in the base and in the image that are likely to be parallel (d1, parallel lines are colored the same), orthogonal (d2, solid lines are orthogonal to dashed lines), symmetric (d3, showing only the projected symmetry axes), or ends of a same rail (d4). It optimizes the orientation of the base plane, thereby reconstructing the object first in wireframe (e) and then in solid (f).*

Here we weight each asymmetry term by the relative curvature difference $curv(p_{\mathbf{n}}, q_{\mathbf{n}})$, which is the difference in curvature of the base curve at the two points $p_{\mathbf{n}}, q_{\mathbf{n}}$ normalized by the greater curvature among the two. This is because two points with similar curvature on the curve are more likely to be symmetric, and hence symmetry with respect to such two points should contribute more to the overall symmetry. We used $\sigma_3 = 0.05$ in our implementation.

Image matching While the user only sketches the base (which is facing the camera), part of the boundary of the cap is usually visible in the image and easy to detect as strong edges. We exploit the matching between the 3D cap defined by the plane normal and creases in the image to increase the robustness of normal estimation.

Given the user-provided 2D rail vector r and the normal $\mathbf{n}$, let $r_{\mathbf{n}}$ be the 3D rail vector that is parallel to $\mathbf{n}$ and whose projection onto the image plane is r. For a point p sampled on the sketched base curve, the point on the cap corresponding to $p_{\mathbf{n}}$ on the base is uniquely defined as $p_{\mathbf{n}} + r_{\mathbf{n}}$, whose projection on the image is denoted as $\bar{p}_{\mathbf{n}}$. Ideally, this projected location should be near to an image crease. The matching objective is defined for a set S_m of point pairs of the form $\{p, q\}$ where p is on the sketched base curve and q is on an image crease,

$$f_m(\mathbf{n}, S_m) = \frac{1}{|S_m|} \sum_{\{p,q\} \in S_m} \exp\left(-\frac{(\bar{p}_{\mathbf{n}} - q)^2}{\sigma_4^2}\right), \qquad (4)$$

where $\sigma_4 = 0.1$ in our implementation.

Putting together The complete formulation of our optimization goal involves a weighted sum of the four objectives described above. Given a collection of various candidate sets for each objective $S = \{S_p, S_o, S_s, S_m\}$, we aim to maximize the fitness function

$$f(\mathbf{n}, S) = w_p f_p(\mathbf{n}, S_p) + w_o f_o(\mathbf{n}, S_o) + w_s f_s(\mathbf{n}, S_s) + w_m f_m(\mathbf{n}, S_m). \qquad (5)$$

Note that, by definition, each of the four objectives is normalized to lie within $[0, 1]$. Empirically we found that the weights $w_p = w_o = 1.0, w_s = 2.0, w_m = 3.0$ works well for all our examples.

4.3. Optimization

To solve the optimization problem, we first need to find the candidate set S. Ideally, we would like to select those candidate pairs of lines or points that are likely to be parallel

(for S_p), orthogonal (for S_o), symmetric (for S_s) or ends of a same rail (for S_m). Examples of these candidate pairs are shown in Figure 3 (d).

We find S by sampling a small set of normals. For each sampled normal direction $\mathbf{n}$ of the base plane, we identify the candidate sets $S^{\mathbf{n}} = \{S_p^{\mathbf{n}}, S_o^{\mathbf{n}}, S_s^{\mathbf{n}}, S_m^{\mathbf{n}}\}$ containing the likely pairs for that base plane. We then pick the normal $\mathbf{n}_0$ that maximizes the fitness cost $f(\mathbf{n}, S^{\mathbf{n}})$, and uses its candidate sets $S = S^{\mathbf{n}_0}$. Then the optimization problem of Equation 5 is solved for $\mathbf{n}$.

Normal sampling While theoretically $\mathbf{n}$ can lie anywhere on the hemisphere of unit vectors with a positive Z component (since base faces towards the camera), the 2D rail vector r given by the user limits our sampling space to only half of a great circle, which is the intersection of the hemisphere with a 3D plane formed by r on the image plane and the camera origin. In practice, we only sample $\mathbf{n}$ along a half of this semicircle since $\mathbf{n}$ points away from r.

We use the following criteria to select candidate pairs $S^{\mathbf{n}}$ for each sampled $\mathbf{n}$. A pair of line segments $\{s,t\} \in S_p^{\mathbf{n}}$ if the 3D angle $\theta(s_{\mathbf{n}}, t_{\mathbf{n}})$ is less than 20 degrees, and $\{s,t\} \in S_o^{\mathbf{n}}$ if $\theta(s_{\mathbf{n}}, t_{\mathbf{n}})$ is more than 70 degrees. A pair of points $\{p,q\} \in S_s^{\mathbf{n}}$ if the relative curvature difference $curv(p_{\mathbf{n}}, q_{\mathbf{n}})$ is less than 0.4, and there are at least 15% of all point pairs $\{p',q'\}$ whose asymmetry with respect to $\{p,q\}$, $asym(p_{\mathbf{n}}, q_{\mathbf{n}}, p'_{\mathbf{n}}, q'_{\mathbf{n}})$, is less than 0.2. For S_m, we first register the projected visible portion of the cap boundary with the image creases using ICP. Then, a pair of curve point p and image point q is selected if q is the registered location for $\bar{p}_n$ and if their distance is smaller than 15 (pixels)

Optimizing for n The fitness function of Equation 5 is highly non-linear. For optimization, we used the KNITRO package [BNW06] due to its robustness and efficiency for solving large and high dimensional problems. For faster convergence, we start with the normal $\mathbf{n}_0$ that maximizes the fitness cost during normal sampling, since it is where the candidates sets $S^{\mathbf{n}}$ are chosen. We have observed that this optimization is very fast in practice, usually taking no more than 2 seconds (16-threaded).

Creating a 3D model Once the plane normal $\mathbf{n}$ is determined, we generate a 3D mesh for the extruded object as follows. First we create the base by triangulating the 3D sampled points $p_{\mathbf{n}}$ on the base plane. The base is duplicated and translated by the 3D rail vector $r_{\mathbf{n}}$ to form the cap. Finally, the side is constructed as a triangle strip between the base and the cap. We can also texture the base and visible parts of the side of the reconstructed extruded object using the original image. The cap face, which is invisible, uses the same texture on the base.

4.4. Validation

We validate our algorithm using a synthetic image rendered from an extruded object with a known geometry, as shown in Figure 4 (top left). The object has a highly non-trivial base (shown in the insert). The top edge of the chair is picked as the rail, and a point on the base is used to fix the depth of the base plane.

To test the effectiveness of the various optimization objectives, we first performed optimization using each of the four objectives (parallelism, orthogonality, symmetry, and image matching) alone. Note that the image fitting objective is most effective for this example, due to the strong creases around the cap. The second most effective objective is symmetry, which captures the partial symmetry in the lower part of the base curve, followed by the orthogonality term that captures the orthogonal relation between the upper and lower parts. The best result is obtained by using all objectives in the optimization (Figure 4 bottom-right).

5. Modeling multiple parts

In addition to modeling a single extruded shape, our tool allows creation of complex objects made up of extruded parts. The user may create each part individually as discussed above. For man-made objects, it is typical to see transformed copies of a same part (e.g., four legs of a chair). To save interaction time, our tools allows the user to conveniently copy and transform from an existing part.

The main challenge in multi-part modeling is determining the relative depths of different parts [CZS*13]. In this work, we utilize simple cues provided by the user in the form of contact points or shared axes. The depths of each part is then resolved using a constrained optimization.

5.1. User interface

To copy an existing part, the user can click on the image to select the part. The part can be translated, reflected, or rotated. To define a transformation, the user first selects an edge of the part as the *axis*. The user can then drag the mouse to translate (in 3D) along the axis direction, reflect by a plane orthogonal to the axis, or rotate around the axis (see Figure 5 (c1,d1)). Our tool then optimizes these parts so that their projections snap to the image edges (Figure 5 (c2,d2)).

The user can specify depth relations among parts in two ways. First, she can designate contact points that are shared by two parts. Given a user-picked 2D location, the corresponding points on the 3D parts are found by ray-intersection with the geometry. In the example of Figure 6 (a), two contact points are specified between the back and arm of the chair. Second, she can constrain two parts or groups of parts to share a common axis. The axis of a single part is aligned with the rails and passes through the center of the base. For a group of parts selected by the user, she can designate any edge of a part as the axis direction, and the axis goes through the centroid of the parts. In the example of Figure 6 (b), the co-axial constraint is set between the table top (single part) and the four legs (parts group).

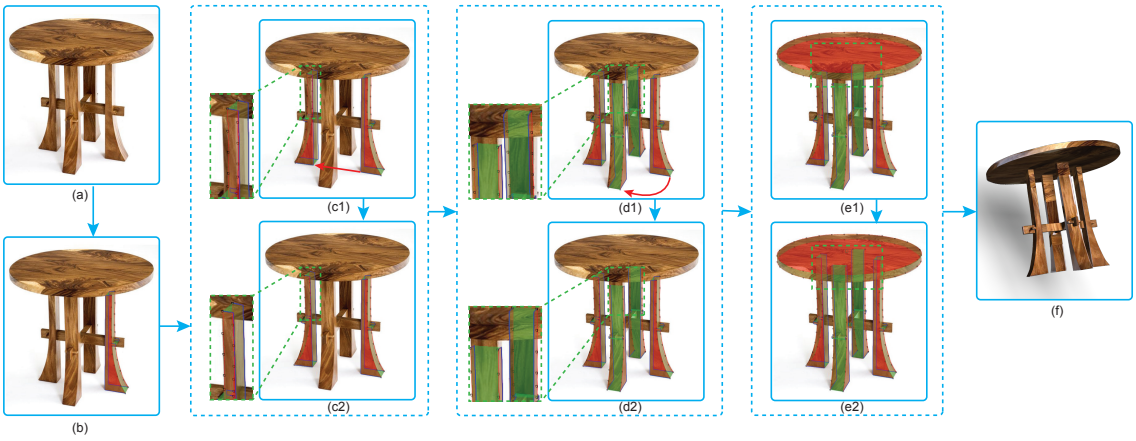

Figure 5: *Modeling a multi-part object. After creating one leg of the table (a,b), the user creates the other legs by translation (c1) and rotation (d1), and the transformed parts are optimized to snap onto image edges (c2,d2). The tool also automatically truncates the hidden upper ends of the legs by the lower plane of the table top (e1,e2).*

5.2. Optimizing transformed parts

The goal of optimizing a transformed part is two-fold. First, the projection of the base and cap outlines should maximally matches image creases. Second, the regularity of the base shape should be preserved. Note that this is similar to the objectives in optimizing a single part from user-sketched base curve, with the only difference that we no longer have an accurate base curve to start with.

We set up and solve a similar optimization problem as in Equation 5, now for variables that include not only the plane normal $\mathbf{n}$, but also rail vector r and base curve points P. We expand the image matching objective f_m to measure the fitting between image creases and the base curve points $p \in P$, in addition to the cap points $\bar{p}_{\mathbf{n}}$. The candidate sets S_p, S_o, S_s are chosen as the same as the original part from which this part is copied. For image matching, the set S_m of candidate point pairs between the base curve and the image is determined by registering the initial locations of P with nearby image creases. Then optimization is performed starting from the initial values of $\mathbf{n}, P, r$ of the transformed part.

5.3. Optimizing all parts

In this final optimization, we adjust the relative placement of the parts to meet the depth constraints provided by the user. As we are considering multiple parts, this also gives us the opportunity to discover global regularity among different parts, such as near-parallel or near-orthogonal pairs of base planes. We use these as constraints to further refine the orientation of each part.

We formulate a constrained optimization problem that maximizes the total fitness of all parts while respecting constraints in depth and orientation. The variables include the base plane normal $\mathbf{n}_i$ and depth d_i for the i-th part. Specifically, we aim to

$$\begin{aligned} &\text{maximize} && \sum_i f(\mathbf{n}_i, S_i) \\ &\text{subject to} && \{C_{para}, C_{orth}, C_{cont}, C_{coax}\}. \end{aligned} \tag{6}$$

The various constraints are defined as follows. For two normals $\mathbf{n}_i, \mathbf{n}_j$ that are near parallel (with an angle smaller than 10 degrees), we add a parallel constraint $\mathbf{n}_i \times \mathbf{n}_j = 0$ to C_{para}. Similarly, if $\mathbf{n}_i, \mathbf{n}_j$ are near orthogonal (with an angle greater than 80 degrees), we add an orthogonal constraint $\mathbf{n}_i \cdot \mathbf{n}_j = 0$ to C_{ortho}. For each contact point specified by the user, we add a constraint to C_{cont} that asks the corresponding points on the two parts to be co-located in 3D. If a co-axial relation is given by the user between an axis v passing through point p and another axis-point pair v', p', we add constraints $v \times v' = 0$ and $(p - p') \times v = 0$ to C_{coax}.

Our formulation naturally gives rise to a two-step optimization process. In the first step, we fix the normals $\mathbf{n}_i$ and solves depths d_i as a constrained equation system. Next we fix d_i and optimizes for $\mathbf{n}_i$. The two steps are iterated until convergence.

6. Results and discussion

We present more results in Figures 7 and 8. We have found that man-made objects, and in particular furniture, is usually made up primarily of extruded parts. These parts often have a complex, and often curved, base shape, which makes them difficult to model using existing interactive modeling tools.

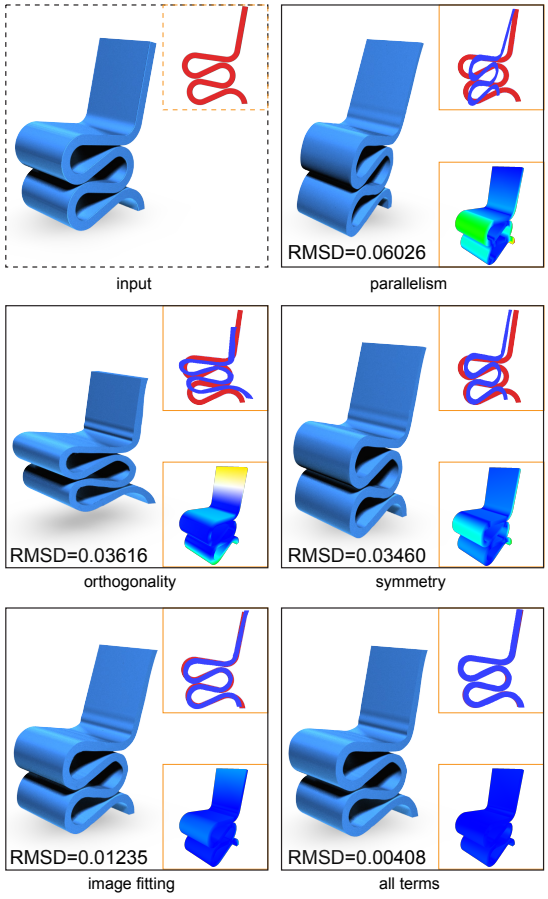

Figure 4: *A synthetic image with a known geometry (top-left), reconstructions using each of the four optimization objectives alone (top-right through bottom left), and the result using all four objectives (bottom right). Top inserts in each image show the optimized base curve (blue) as viewed on the base plane overlayed on the ground truth (red), and bottom inserts plot the distance (in heat color) between the ground truth and reconstructed geometry. The reconstruction error is reported in RMSD as fraction of the longest dimension of the model.*

On the other hand, these base shapes usually have enough regularity for our tool to produce an accurate model with only a small amount of user input.

All experiments were carried out on a personal computer with a 2.27GHz Intel E5520 CPU and 16GB memory. Optimizing for a single part takes between 1 second (e.g., the chair in Figure 3) to 5 seconds (e.g., the synthetic example in Figure 4). Optimizing for multiple components, including adjusting transformed parts, ranges between 2 seconds (e.g., the last step in Figure 1) to 5 seconds (e.g., the last step of Figure 5). Table 1 reports the detailed timing for various types of interactions, camera calibration, and computations for a few selected examples. All processes are 16-threaded.

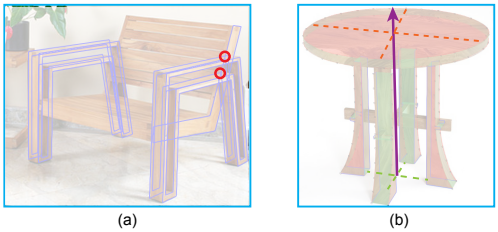

Figure 6: *Two kinds of depth cues: contact points between back and arm of the chair (a), and co-axial relation between table top and the group of four table legs (b).*

7. Conclusion and discussion

In this paper, we present a new interactive tool for objects made up of extruded parts. Unlike previous tools, we allow the base curve of the extruded object to have non-trivial, curved shapes. Our optimization-based solution combines user inputs, image edges, and regularities both within each part and among multiple parts. The method is validated on a synthetic example and tested on a suite of real-world images.

Limitations Our tool makes several assumptions that may not hold for some images. First, both the cap and the base of the extruded object must be partly visible. Hence we cannot model facades, like the one in Figure 9 (a), where only the base is visible. Second, we cannot handle irregular base curves that lack any parallelism, orthogonality, or partial symmetry, such as the table-top in Figure 9 (b).

Extensions A possible extension of our method is to model more general extruded objects whose rails are not orthogonal to the base (e.g., a slanted tube) or curved (e.g., a bent tube), or whose cross-section shapes undergo scaling changes (e.g., a pyramid or frustum). Although the current method takes advantage of the orthogonality of the rail in optimizing the orientation of the base plane, this can be possibly replaced by more user input and prior knowledge of the shape. For example, we can ask the user to pick not one, but two rails for modeling a pyramid or frustum object.

Acknowledgement: We would like to thank the anonymous reviewers for their valuable comments. This work was supported by the National Basic Research Project of China (Project Number 2011CB302202), the Natural Science Foundation of China (Project Number 61120106007), the National High Technology Research & Development Program of China (Project Number 2012AA011802), Research Grant of Beijing Higher Institution Engineering Research Center, Tsinghua University Initiative Scientific Research Program, and NSF (USA) grants (IIS-0846072 and IIS-1302200).

Figure 8: *A rendered scene of all furniture created by our tool.*

Model	Calib.	Sketch base	Pick rail	Multi-part	PC
Bench	13	22	2	9	2.86
Chair	15	17	1	0	0.96
Table	16	21	5	17	13.95
Bridge	15	32	6	3	3.38
Bookshelf	39	31	3	9	9.63

Table 1: *User interaction time (in seconds) for camera calibration, sketching base curves, picking rails, and manipulating parts in a few examples (Bench of Figure 1, Chair of Figure 3, Table of Figure 5, and Bridge and Bookshelf of Figure 7). Computation time for each example is shown in the last column.*

References

[BNW06] BYRD R. H., NOCEDAL J., WALTZ R. A.: Knitro: An integrated package for nonlinear optimization. In *Large Scale Nonlinear Optimization* (2006), Springer Verlag, pp. 35–59. 6

[CGZZ13] CHEN X., GUO Y., ZHOU B., ZHAO Q.: Deformable model for estimating clothed and naked human shapes from a single image. *The Visual Computer 29*, 11 (2013), 1187–1196. 3

[Che09] CHENG M.-M.: Curve structure extraction for cartoon images, 2009. 4

[CKX*08] CHEN X., KANG S. B., XU Y.-Q., DORSEY J., SHUM H.-Y.: Sketching reality: Realistic interpretation of architectural designs. *ACM Trans. Graph. 27*, 2 (May 2008), 11:1–11:15. 3

[CZS*13] CHEN T., ZHU Z., SHAMIR A., HU S.-M., COHEN-OR D.: 3-sweep: Extracting editable objects from a single photo. *ACM Trans. Graph. 32*, 6 (Nov. 2013), 195:1–195:10. 1, 3, 6

[DABP14] DOLLÁR P., APPEL R., BELONGIE S., PERONA P.: Fast feature pyramids for object detection. *PAMI* (2014). 4

[ERB*12] EITZ M., RICHTER R., BOUBEKEUR T., HILDEBRAND K., ALEXA M.: Sketch-based shape retrieval. *ACM Trans. Graph. (Proc. SIGGRAPH) 31*, 4 (2012), 31:1–31:10. 3

[FWZQ13] FANG T., WANG Z., ZHANG H., QUAN L.: Image-based modeling of unwrappable facades. *IEEE Transactions on Visualization and Computer Graphics 19*, 10 (2013), 1720–1731. 3

[HZ03] HARTLEY R., ZISSERMAN A.: *Multiple View Geometry in Computer Vision*, 2 ed. Cambridge University Press, New York, NY, USA, 2003. 3

[JTC09] JIANG N., TAN P., CHEONG L.-F.: Symmetric architecture modeling with a single image. *ACM Trans. Graph. 28*, 5 (Dec. 2009), 113:1–113:8. 1, 3

[KW11] KELLY T., WONKA P.: Interactive architectural modeling with procedural extrusions. *ACM Trans. Graph. 30*, 2 (Apr. 2011), 14:1–14:15. 3

[LLT07] LI Z., LIU J., TANG X.: A closed-form solution to 3d

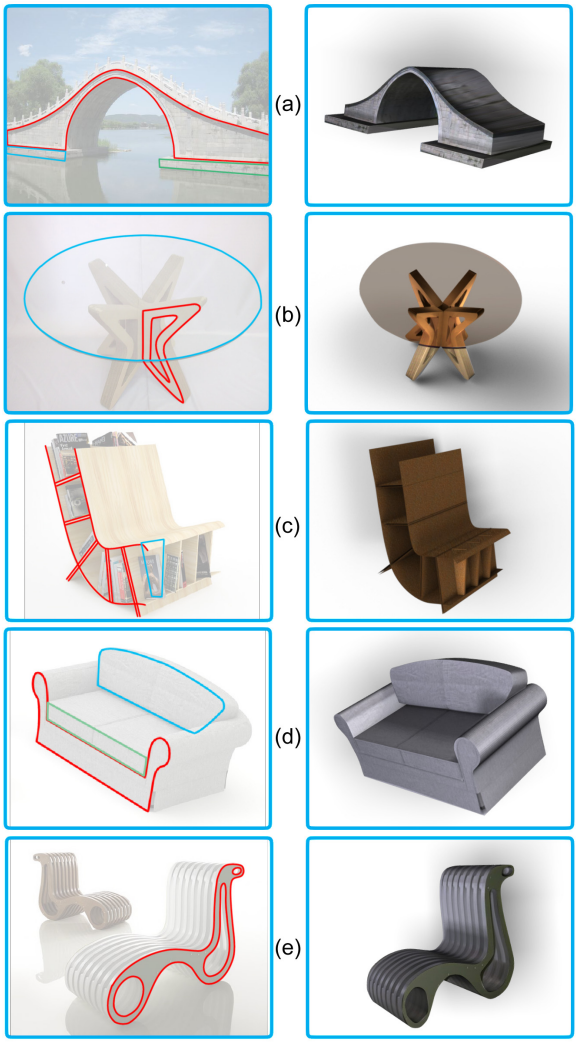

Figure 7: *More results of our tool, showing the image with the user-drawn base curves (left) and textured geometry (right).*

(a) (b)

Figure 9: *Some failure cases.*

reconstruction of piecewise planar objects from single images. In *CVPR* (2007). 4

[LS07] LIPSON H., SHPITALNI M.: Optimization-based reconstruction of a 3d object from a single freehand line drawing. In *ACM SIGGRAPH 2007 Courses* (2007), SIGGRAPH '07. 3

[LSMI10] LAU M., SAUL G., MITANI J., IGARASHI T.: Modeling-in-context: User design of complementary objects with a single photo. In *Proceedings of the Seventh Sketch-Based Interfaces and Modeling Symposium* (Aire-la-Ville, Switzerland, Switzerland, 2010), SBIM '10, Eurographics Association, pp. 17–24. 3

[MGP07] MITRA N. J., GUIBAS L., PAULY M.: Symmetrization. *ACM Transactions on Graphics (SIGGRAPH) 26*, 3 (2007), #63, 1–8. 4

[OCDD01] OH B. M., CHEN M., DORSEY J., DURAND F.: Image-based modeling and photo editing. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques* (2001), SIGGRAPH '01, pp. 433–442. 2

[OUP*11] ÖZTIRELI A. C., UYUMAZ U., POPA T., SHEFFER A., GROSS M.: 3d modeling with a symmetric sketch. EG. 3

[RDI10] RIVERS A., DURAND F., IGARASHI T.: 3d modeling with silhouettes. *ACM Trans. Graph. 29*, 4 (July 2010), 109:1–109:8. 3

[Sug86] SUGIHARA K.: *Machine Interpretation of Line Drawings*. Artificial Intelligence Series. MIT Press, 1986. 3

[TBSR04] TSANG S., BALAKRISHNAN R., SINGH K., RANJAN A.: A suggestive interface for image guided 3d sketching. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2004), CHI '04, pp. 591–598. 2

[VM00] VARLEY P. A. C., MARTIN R. R.: A system for constructing boundary representation solid models from a two-dimensional sketch. In *GMP* (2000), pp. 13–32. 3

[VM02] VARLEY P. A. C., MARTIN R. R.: Estimating depth from line drawing. In *Proceedings of the Seventh ACM Symposium on Solid Modeling and Applications* (2002), SMA '02, pp. 180–191. 4

[WCLT09] WANG Y., CHEN Y., LIU J., TANG X.: 3d reconstruction of curved objects from single 2d line drawings. In *CVPR* (2009), pp. 1834–1841. 3

[WSB05] WILCZKOWIAK M., STURM P. F., BOYER E.: Using geometric constraints through parallelepipeds for calibration and 3d modeling. *IEEE Trans. Pattern Anal. Mach. Intell. 27*, 2 (2005), 194–207. 3

[XCF*13] XU K., CHEN K., FU H., SUN W.-L., HU S.-M.: Sketch2scene: Sketch-based co-retrieval and co-placement of 3d models. *ACM Transactions on Graphics 32*, 4 (2013), 123:1–123:12. 3

[XCS*14] XU B., CHANG W., SHEFFER A., BOUSSEAU A., MCCRAE J., SINGH K.: True2form: 3d curve networks from 2d sketches via selective regularization. *Transactions on Graphics (Proc. SIGGRAPH 2014) 33*, 4 (2014). 3

[XLT12] XUE T., LIU J., TANG X.: 3-d modeling from a single view of a symmetric object. *IEEE Transactions on Image Processing 21*, 9 (2012), 4180–4189. 1, 3

[XZZ*11] XU K., ZHENG H., ZHANG H., COHEN-OR D., LIU L., XIONG Y.: Photo-inspired model-driven 3d object modeling. *ACM Trans. Graph. 30*, 4 (July 2011), 80:1–80:10. 3

[ZCC*12] ZHENG Y., CHEN X., CHENG M.-M., ZHOU K., HU S.-M., MITRA N. J.: Interactive images: Cuboid proxies for smart image manipulation. *ACM Trans. Graph. 31*, 4 (July 2012), 99:1–99:11. 1, 3

[ZLL12] ZOU C., LIU J., LIU J.: Precise 3d reconstruction from a single image. In *ACCV (4)* (2012), pp. 271–282. 3