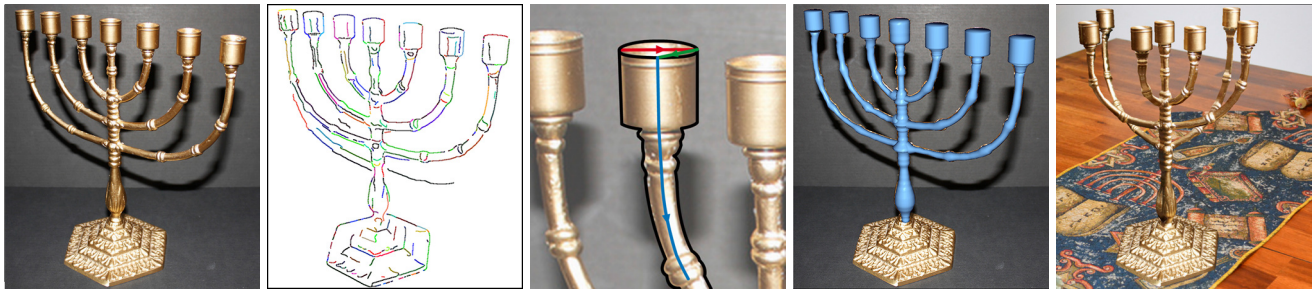
3-Sweep: Extracting Editable Objects from a Single Photo

Tao Chen[1,2,3*] Zhe Zhu[1] Ariel Shamir[3] Shi-Min Hu[1] Daniel Cohen-Or[2]
[1]TNList, Department of Computer Science and Technology, Tsinghua University
[2]Tel Aviv University [3]The Interdisciplinary Center



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

Figure 1: *3-Sweep Object Extraction. (a) Input image. (b) Extracted edges. (c) 3-Sweep modeling of one component of the object. (d) The full extracted 3D model. (e) Editing the model by rotating each arm in a different direction, and pasting onto a new background. The base of the object is transferred by alpha matting and compositing.*

Abstract

We introduce an interactive technique for manipulating simple 3D shapes based on extracting them from a single photograph. Such extraction requires understanding of the components of the shape, their projections, and relations. These simple cognitive tasks for humans are particularly difficult for automatic algorithms. Thus, our approach combines the cognitive abilities of humans with the computational accuracy of the machine to solve this problem. Our technique provides the user the means to quickly create editable 3D parts— human assistance implicitly segments a complex object into its components, and positions them in space. In our interface, three strokes are used to generate a 3D component that snaps to the shape's outline in the photograph, where each stroke defines one dimension of the component. The computer reshapes the component to fit the image of the object in the photograph as well as to satisfy various inferred geometric constraints imposed by its global 3D structure. We show that with this intelligent interactive modeling tool, the daunting task of object extraction is made simple. Once the 3D object has been extracted, it can be quickly edited and placed back into photos or 3D scenes, permitting object-driven photo editing tasks which are impossible to perform in image-space. We show several examples and present a user study illustrating the usefulness of our technique.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Geometric algorithms, languages, and systems;

Keywords: Interactive techniques, Photo manipulation, Image processing

Links: DL PDF

*This work was performed while Tao Chen was a postdoctoral researcher at TAU and IDC, Israel.

1 Introduction

Extracting three dimensional objects from a single photo is still a long way from reality given the current state of technology, as it involves numerous complex tasks: the target object must be separated from its background, and its 3D pose, shape and structure should be recognized from its projection. These tasks are difficult, and even ill-posed, since they require some degree of semantic understanding of the object. To alleviate this problem, complex 3D models can be partitioned into simpler parts, but identifying object parts also requires further semantic understanding and is difficult to perform automatically. Moreover, having decomposed a 3D shape into parts, the relations between these parts should also be understood and maintained in the final composition.

In this paper we present an interactive technique to extract 3D man-made objects from a single photograph, leveraging the strengths of both humans and computers (see Figure 1). Human perceptual abilities are used to partition, recognize and position shape parts, using a very simple interface based on triplets of strokes, while the computer performs tasks which are computationally intensive or require accuracy. The final object model produced by our method includes its geometry and structure, as well as some of its semantics. This allows the extracted model to be readily available for intelligent editing, which maintains the shape's semantics.

Our approach is based on the observation that many man-made objects can be decomposed into simpler parts that can be represented by a generalized cylinder, cuboid or similar primitives. The key idea of our method is to provide the user with an interactive tool to guide the creation of 3D editable primitives. The tool is based on a rather simple modeling gesture we call *3-sweep*. This gesture allows the user to explicitly define the three dimensions of the primitive using three sweeps. The first two sweeps define the first and second dimension of a 2D profile and the third, longer, sweep is used to define the main curved axis of the primitive.

As the user sweeps the primitive, the program dynamically adjusts the progressive profile by sensing the pictorial context in the photograph and automatically snapping to it. Using such 3-sweep operations, the user can model 3D parts consistent with the object in the photograph, while the computer automatically maintains global constraints linking it to other primitives comprising the object.

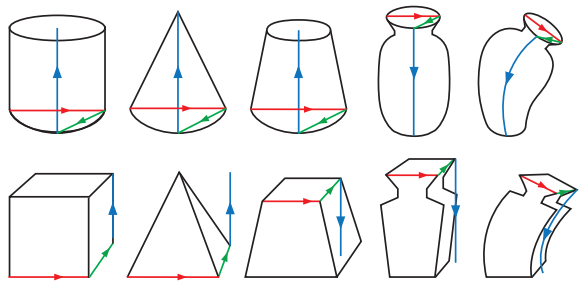


Figure 2: *3-sweep paradigm used to define general cylinders and cuboids.*

Using 3-sweep technology, non-professionals can extract 3D objects from photographs. These objects can then be used to build a new 3D scene, or to alter the original image by changing the objects or their parts in 3D and pasting them back into the photo.

We demonstrate our ideas using various examples, and present a user study to evaluate the usability and efficiency of our tool.

2 Related Work

Part-based Modeling. Our work is inspired by [Shtof et al. 2013] that presented a tool for modeling simple 3D objects from sketches. Like them, we use geo-semantic constraints, such as parallelism and collinearity, to define the semantic and geometric relations between the primitive parts of the final 3D model. However, their approach is geared towards sketches; it relies on pre-segmentation of the sketches and manual curve classification. Their interface requires the user to choose and drop a primitive over the relevant part of the sketch, an approach which demands non trivial optimization to modify and fit the primitive to the sketch contours. In contrast, our 3-sweep approach directly suggests the shape of the parts and provides instantaneous snapping to the contours, offering direct and simple 3D modeling. An accompanying video shows a side-by-side comparison of modeling sessions using our tool and the one in [Shtof et al. 2013].

Other part-based snapping techniques have been used for modeling 3D objects from sketches. A gestural interface based on line drawings of primitives is proposed by Zeleznik et al. [1996]. Schmidt et al. [2005] use BlobTree for sketch-based modeling. Complex models with arbitrary topology can be quickly created. Gingold et al. [2009] provide an interface to generate 3D models from 2D drawings by manually placing 3D primitives. Tsang et al. [2004] use a guiding image to assist sketch-based modeling. The user's input curves snap to the image and suggestively be completed by a curve database.

3D Modeling from a single photo. Images have always been an important resource for 3D modeling. Some techniques model shapes from multiple images [Seitz et al. 2006; Tan et al. 2007; Snavely 2011]. We focus on modeling from a single photograph, where the task is more challenging since there is ambiguity in the observed geometry. In the following, we classify single-photo methods into two categories.

Methods in the first category require extensive manual interaction, and are more time consuming compared to 3-Sweep. They either require a complete sketch or tedious labeling before the machine takes full control of an optimization process, while in our method, user guidance and automatic snapping are interlaced. Some of these methods have limited application ranges, e.g. [Debevec et al. 1996; Jiang et al. 2009; Arikan et al. 2013] consider architectural images, while others are restricted to non-oblique views and extrusion/inflation [Olsen and Samavati 2010], or do not guarantee to make the resulting model a precise match to the input image [Schmidt et al. 2009; Andre and Saito 2011]. Yet others require extensive interaction and annotation, to allow more complicated shapes. Terzopoulos et al. [1988] apply axisymmetry and image-based constraints on deformable object models to infer 3D structures from images. Oh et al. [2001] permit annotation of depth and layer information in a single image and yield impressive image editing results at the scene level. Russell et al. [2009] use a manually annotated database of 3D scenes to assist recovering scene-level geometry and camera pose.

The other category focuses on automatic fitting and optimization to match the image or sketch guidance [Binford 1971; Zhang et al. 2010; Metaxas 1996; Mille et al. 2008]. These are basically 2D methods, either generating 2D models, or 3D models by extrusion, while our method can directly generate oblique shapes in 3D space (see our telescope and menorah examples). [Bolle and Vemuri 1991] summarizes early work on using generalized cylinder for 3D surface reconstruction in computer vision. Most methods fail for such examples as they rely on clear edges or region color, which are not always present, and part edges can interfere with each other. Other methods such as [Xue et al. 2011; Oswald et al. 2012] rely heavily on symmetry and smoothness.

Our work is closely related to the work of Xu et al. [2011] that also uses semantic geometric constraints. However, their method relies on matching and warping an existing 3D object to the object in the photograph. They strongly depend on the existence, and retrieval, of a similar 3D shape, while we reconstruct an object out of simple parts that are interactively defined.

Sweep-based Modeling. Sweep based models have been studied extensively in computer-aided design. Starting from Choi and Lee [1990] that model sweep surfaces by using coordinate transformations and blending, up to Swirling-Sweepers [Angelidis et al. 2004], a volume preserving modeling technique capable of arbitrary stretching while avoiding self-intersection, and an implicit modeling method based on topology-free 2D sweep template proposed by Schmidt and Wyvill [2005]. While we cannot report all CAD work aiming at modeling generalized primitives, to our knowledge, none of these methods have been applied to modeling from photographs, nor they have paired sweeping with snapping.

Semantic Constraints. Gal et al. [2009] introduced a 3D deformation method which preserves some semantic constraints relating the object's parts. Such geo-semantic constraints [Zheng et al. 2011; Merrell and Manocha 2011] have been shown to help to quickly edit or deform man-made models [Xu et al. 2011; Xu et al. 2012]. Li et al. [2011] and Shtof et al. [2013] reconstruct 3D shapes while simultaneously inferring global mutual geo-semantic relations between their parts. Benko et al. [2002] propose to automatically infer constraints in a fitting process of reverse engineering. In our work, we adopt such geo-semantic constraint inference to assist in modeling man-made objects.

Object-Level Image Editing. Unlike traditional image-based editing, object-based editing allows high-level operations. Operating at the object-level requires extensive user interaction [Eitz et al. 2007; Cheng et al. 2010] or massive data collection [Lalonde et al. 2007; Goldberg et al. 2012]. Barrett et al. [2002] use wrapping as a basis for object-level editing, which is restricted to 3D rotation. Zhou et al. [2010] fit a semantic model of a human to an image, allowing object-based manipulation of a human figure in photographs. Recently, Zheng et al. [2012] propose use of cuboid proxies for semantic image editing. Man-made objects are modeled by a set of cuboid proxies, possibly together with some geometric relations or constraints, allowing their manipulation in the photo.

Our method achieves similar image manipulations but permits a larger variety of more complex man-made models with a richer set of geo-semantic constraints. We can also recover a full 3D model of the object rather than just a proxy, and support more shapes than just cuboids.

3 Overview

Our interactive modeling approach takes as input a single photo such as the one in Figure 1(a). Our goal is to extract a 3D model whose projection exactly matches the object in the image. Using the *3-sweep* modeling technique, the user constructs the whole object from parts. The user implicitly decomposes the object into simple parts, typically which are semantically meaningful. Such decomposition is both easy and intuitive for users, but provides the computer with significant information for reconstructing a coherent 3D man made object from its projection. The parts are expected to have typical geometric relations that can be exploited to guide the composition of the whole object.

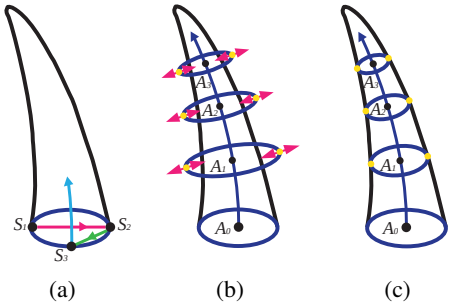
Although the user interacts with the given photo, 3-Sweep relies on snapping primitives to object outlines created from image edges. To extract image edges and build candidate object outlines, we adopt a method for hierarchical edge feature extraction based on spectral clustering [Arbelaez et al. 2011]. We then apply a technique to link the detected edge pixels into continuous point sequences [Cheng 2009], each shown in a different color in Figure 1(b). An edge orientation computed over a 5×5 neighborhood is associated with each edge pixel.

To create a single part, the user interactively fits a 3D primitive into the given photo. This operation is not trivial since the photo lacks the third dimension and fitting is inherently ambiguous. The challenge is to provide the interactive means to disambiguate such fitting. Our 3-sweep technique requires the user to generate a 3D model that roughly approximates the target part, and snaps to the extracted outline of the object.

The 3D approximate part is defined by generating a 2D profile of the part using 2 strokes and then defining its main axis using a third stroke, which is a straight or curved axis (see Figure 2). Defining the profile as well as the sweeping operation are simple tasks since they do not demand accuracy. The profile dimensions are guided by the object's outlines. While sweeping, the 3D extent of the part is also defined by snapping to these outlines. Thus, the part need only be sketched quickly and casually by the user. Figure 1(c) shows the result of sweeping along the tubes of the menorah. We elaborate on the 3-sweep operation in Section 4. To compensate for perspective distortion, during this process, the camera's angle of view is estimated.

As the model parts are constructed, geometric relations between them serve (i) to assist in disambiguating and defining the depth dimension and (ii) to optimize the positioning of the parts. These geometric relations include parallelism, orthogonality, collinearity and coplanarity. We optimize satisfaction of these geo-semantic constraints while taking into account the snapping of the 3D geometry to the object's outlines and the user's sweeping input. A complete model with geo-semantic relations is shown in Figure 1(d). These geo-semantic relations not only help define the 3D model, but are also recorded as part of the 3D representation allowing later smart editing of the 3D model, as demonstrated in Figure 1(e) and other figures in the paper.

Our interface also supports several operation for more effective modeling. For example, the user can constrain the parts to have uniform or linearly changing radii, or copy and paste similar parts. Although many geo-semantic constraints are automatically inferred, the user can also manually specify constraints between selected parts. The user can also adjust the vertical angle of view (45° by default) for scenes captured by tilted cameras. Details are discussed later.



(a) (b) (c)

Figure 3: *Modeling a primitive by defining a 2D profile and sweeping it along the main axis of the object (see text).*

4 Single Primitive Fitting

In this section, we first describe the 3-sweep technique for generalized cylinders and then briefly show how it extends to the simpler case of generalized cuboids. Figure 2 and Figure 3 illustrate the 3 strokes used with red and green arrows, and blue curved arrows.

Profile In the first stage, the user draws the 2D profile of the generalized cylinder, usually at one end of the shape. This is illustrated in Figure 3, where black curves are outlines detected in the input image. The task is to draw a 2D profile correctly oriented in 3D. This can be regarded as positioning a disk in 3D by drawing its projection in 2D. To simplify this task, we assume that the disk is a circle, thus reducing the number of unknown parameters. Later, the circular disk can be warped into an elliptical disk based on the 3D reconstruction. The drawing of a circular disk is accomplished by drawing two straight lines S_1S_2 and S_2S_3 over the image, as shown in Figure 3(a). The first line defines the major diameter of the disk, and then the second line is dragged to the end of the minor diameter. This forms an ellipse in image space that matches the projection of a circular disk: see the dark blue circle in Figure 3(a). The depth of the disk is set to 0. The normal direction and radius of the disk are assigned according to the length and orientation of the two diameters of the elliptical projection.

Sweeping After completing the base profile, the user sweeps it along a curve that approximates the main axis of the 3D part. In general, this curve should be perpendicular to the profile of the 3D primitive (see blue arrow in Figure 3 (a)). As the curve is drawn, copies of the profile are placed along the curve, and each of them is snapped to the object's outline.

During drawing, the axis curve is sampled in image space at a uniform spacing of five pixels to produce sample points $A_0, \ldots, A_N$. At each sampled point A_i, a copy of the profile is centered around the curve. Its normal is aligned with the orientation of the curve at A_i, and its diameter is adjusted to meet the object's outline. Together, the adjusted copies of the profile form a discrete set of slices along the generalized cylinder, as shown in Figure 3(c).

At each point A_i, we first copy the profile from A_{i-1} and translate it to A_i. Then we rotate it to take into account the bending of the curve. We then consider the two ends of the major axis (yellow points in Figure 3 (b)) of the profile, denoted by p_i^0, p_i^1. For each contour point p_i^j, $j \in [0,1]$ we cast a 2D ray from point A_i along the major axis, seeking an intersection with an image outline. Finding the correct intersection of the ray with an image outline is somewhat challenging. The image may contain many edges in the vicinity of the new profile. The closest one is not necessarily the correct one, e.g. when hitting occluding edges. In other cases, the correct

edges may be missing altogether. To deal with such cases, we first limit the search for an intersection to a fixed range, which limits the major axis of adjacent profiles not to vary by more than 20% in length. Secondly, we search for an intersecting outline that is close to perpendicular to the ray. If the angle between the ray and the outline is larger than $\pi/3$, the candidate intersection is discarded. Although this method cannot guarantee to find the intersections, the subsequent profile propagation step can tolerate a limited number of missing intersections.

When an intersection point is found, we snap the contour point p_i^j to it. If both contour points of the profile are snapped, we adjust the location of A_i to lie at their midpoint. If only one side is successfully snapped, we mirror the length of this side to the other side and move the other contour point respectively. Lastly, if neither contour points is snapped, the size of the previous profile is retained.

Other Primitives Generalized cuboids are modeled in a similar way to generalized cylinders. The main difference lies in the first stage of modeling the profile. The two strokes that define the profile of a cuboid follow the two edges of the base of the cuboid instead of the diameters of the disk, as shown by the red and green lines in the bottom row of Figure 2. Simpler primitives such as spheroids or simple cubes are also supported by direct modeling in our system.

The above modeling steps closely follow user gestures, especially when modeling the profile. This provides more intelligent understanding of the shape but is less accurate. Therefore, after modeling each primitive, we apply a post-snapping stage to better fit the primitive to the image as well as correct the view. We search for small transformations ($\pm 10\%$ of primitive size) and changes of vertical angle of view ($\pm 10^\circ$) that create a better fit of the primitive's projection to the edge curves that were snapped to in the editing process.

In many cases, the modeled object has special properties that can be used as priors to constrain the modeling. For example, if we know that a given part has a straight spine, we can constrain the sweep to progress along a straight line. Similarly, we can constrain the sweep to preserve a constant or linearly changing profile radius. In this case, the detected radii are averaged or fitted to a line along the sweep. We can also constrain the profile to be a square or a circle. In fact, a single primitive can contain segments with different constraints: it can start with a straight axis and then bend, or use a constant radius only in a specific part. Such constraints are extremely helpful when the edge detection provides poor results.

To further assist in modeling interaction, we also provide a copy and paste tool. The user can drag a selected part that is already snapped over to a new location in the image and snap it again in the new position. While copying, the user can rotate, scale, or flip the part.

5 Inter-part Optimization

The technique described above generates parts that fit the object outlines. The positions of these parts in 3D are still ambiguous and inaccurate. However, as these parts are components of a coherent man-made object, semantic geometric relations often exist between them. Constraining the shape to satisfy such relations allows the modeling of meaningful shapes [Gal et al. 2009; Zheng et al. 2011; Li et al. 2011; Shtof et al. 2013].

Direct global optimization of the positions of parts while considering their geo-semantic relations is computationally intensive and can get trapped in local minima, since each component has many degrees of freedom. In our setting, the modeled components are also constrained to agree with outlines of the image, and this can significantly reduce the degrees of freedom for each part. By considering the image constraints, the dimensionality of the optimization space can be lowered and local minima avoided. In the following, we describe how we simplify the general problem and solve a not-too-difficult optimization problem to ensure geo-semantic constraints are satisfied between the 3-swept parts.

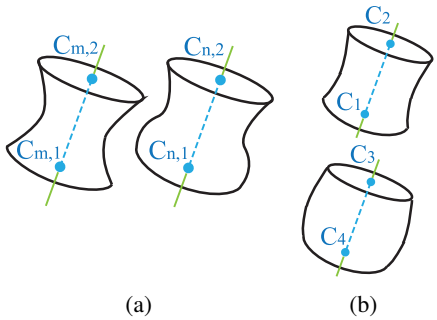


Figure 4: *Inferring geo-semantic constraints. (a) Parallelism. (b) Collinear axis endpoints.*

The key idea is that by fixing the projection of a part, its position and orientation can be determined by one or two depth values only. We first describe the method for simple parts that can be modeled by a single parameter, namely parts which were modeled using a *straight* axis. General cylinders and cuboids with curved axes will later be approximated by two arbitrarily-connected straight axis primitives at the start and end of the shape.

Determining straight shapes by unarivariate functions. The position and orientation of a generalized cylinder i with straight-axis can be determined by two points we call *anchors*, $C_{i,1}$ and $C_{i,2}$, on its main axis (see Figure 4). Similarly, a cuboid part can be represented by six anchors $\mathbf{C}_{i,j},\ j \in [1,6]$ positioned at the center of each face. Every opposite pair of anchors defines one main axis of the cuboid. Even though four anchors are enough to fix the position and orientation of a cuboid, we use six to simplify attaching various geo-semantic constraints to such parts.

The user defines a 3D part i using three strokes for the three dimensions, which we utilize to define a local 3D orthogonal coordinate system for the part. First, we define the origin of the coordinate system at a reference point $\mathbf{R_i}$ on the part's projection. For a cuboid part, we pick the point connecting the first and second of the user's strokes, and for a cylinder we pick the point connecting the second and third strokes. Due to the internal orthogonality of the straight part, the profile of the part is perpendicular to the main axis. Therefore, we can use the endpoints of the user's strokes (after snapping them to the image) to define three points that together with $\mathbf{R_i}$ create an orthogonal system (orange points and lines in Figure 5). Note that this coordinate system is defined in camera coordinates. The x and y values of the end points are determined by the projection and their depth values can be found as a function of z_i, the z value of $\mathbf{R_i}$, by using three orthogonality constraint equations.

Next, the positions of the anchor points $\mathbf{C}_{i,j}$ in world coordinates are defined using the local orthogonal axes, giving the structure of part i. Since the local axes depend only on the depth value z_i of the point $\mathbf{R_i}$, we can parameterize the positions of $\mathbf{C}_{i,j}$ as a function of z_i: $\mathbf{C}_{i,j} = \mathbf{F}_{i,j}(z_i)$: the position and orientation of the whole part become a function of a single unknown z_i. $\mathbf{F_{i,j}}$ has the form $F_{i,j}(z_i) = b/(a(z_i + v))$ for each coordinate component, where a depends only on the x and y coordinates of the endpoints of the local axes, and b, v are decided by perspective parameters. They are different for each axis endpoint and for each coordinate component (see Appendix A).

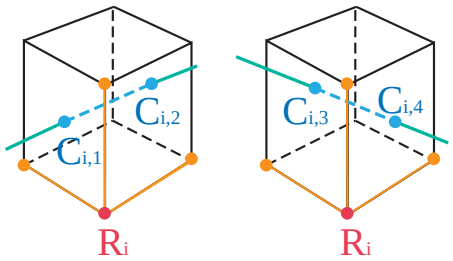


Figure 5: *Determining coordinates $\mathbf{C}_{i,j}$ for axis endpoints of a cuboid from the depth value z_i of the reference point $\mathbf{R_i}$.*

Defining geo-semantic constraints. We use the anchor points to define geo-semantic relations between parts. Specifically, we support six types of constraints: parallelism, orthogonality, collinear axis endpoints, overlapping axis endpoints, coplanar axis endpoints and coplanar axes. During modeling, for each type, we test whether a pair of components is close to satisfying one of the above geo-semantic constraints, and if so, we add the constraint to our system. For example, for two cylinders with indices m and n, if the angle between vector $(\mathbf{C}_{m,1}-\mathbf{C}_{m,2})$ and $(\mathbf{C}_{n,1}-\mathbf{C}_{n,2})$ is less than 15°, we add a parallelism constraint $(\mathbf{C}_{m,1}-\mathbf{C}_{m,2})\times(\mathbf{C}_{n,1}-\mathbf{C}_{n,2})=0$ to the system of constraints. Similarly if any three of the four anchors of two cylinders form a triangle containing an angle larger than 170°, we add a collinear axes constraint: $(\mathbf{C}_1-\mathbf{C}_2)\times(\mathbf{C}_1-\mathbf{C}_3)=0$ (see Figure 4). Internal constraints such as orthogonality and concentricity of a cuboid's axes are also added to the system. Finally, we allow the user to manually enforce or revoke any constraint for selected primitive parts.

Establishing an objective function. Suppose we have found p geo-semantic constraints G_k for a set of n components. Together with the objective function for fitting the image outline, we define the following optimization system:

$$\text{minimize } E=\sum_{i=1}^{n} w_i\Big(\sum_{j=1}^{m_i}||\mathbf{C}_{i,j}-\mathbf{F}_{i,j}(z_i)||^2\Big) \tag{1}$$

$$\text{subject to} \quad G_k(\mathbf{C}_{1,1},...,\mathbf{C}_{n,m_n}), \quad k=1,...,p, \tag{2}$$

where m_i is the number of axes of the ith primitive part. We add weights w_i proportional to the radius of the base profile of each part and the length of its axis. Larger parts have more impact on the solution since typically larger parts are modeled more accurately. Intuitively, the first equation tries to fit the part's geometry ($\mathbf{C}_{i,j}$) to the image outline and the user's gestures, while the second set of equations imposes the geo-semantic constraints.

Two steps solution. Solving for $\mathbf{C}_{i,j}$ and z_i together is a non-linear non-convex optimization problem with non-linear constraints. Directly solving such a system without becoming trapped in a local minimum is very difficult. Hence, we decompose the solution into a two step procedure. The first step tries to find a good initial position for all parts at once, by changing only their depths (governed by z_i) to meet the geo-semantic constraints. In the second step, the full system is solved, allowing the shapes of the parts ($\mathbf{C}_{i,j}$) to change as well.

In first step, we modify the soft constraint in Equation (1) to a hard one, and replace $\mathbf{C}_{i,j}$ by $F_{i,j}(z_i)$ in all equations. This means Equation (1) is trivially true and we are left with just the constraints in Equation (2). In effect, this means we fix the projection and find the optimal z_i meeting the geo-semantic constraints. This reduces the number of variables to n (z_i, $1\le i\le n$) and changes Equation (2) into an over-determined system, where each equation only contains two different variables. We find the least squares solution $\bar{z}_i$ by the conjugate gradient method, with all z_i values initialized to 0.

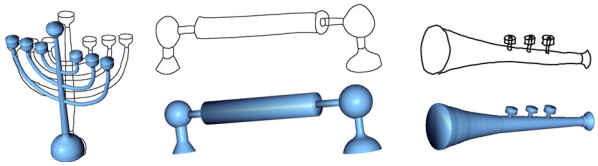


Figure 11: *Modeling from sketches. Input sketches are taken from [Shtof et al. 2013].*

This first step provides a good initialization to find the optimal solution for $\mathbf{C}_{i,j}$, which should be close to $\mathbf{F}_{i,j}(\bar{z}_i)$, requiring only small inconsistencies to be fixed with the geo-semantic constraints. Hence, in the second step, we carry out full optimization of Equation (1) with the set of constraints in Equation (2) by an augmented Lagrangian method. Both steps are fast, and we are able to avoid local minima by better initialization provided by the first step. This permits optimization to be carried out at interactive speed (see the accompanying video). Note that the nonlinearity of $\mathbf{F}_{i,j}()$ arises due to the assumption of a perspective projection. However, we can linearly approximate this projection since we assume the change in z_i is small. This further increases the speed and stability of our solution.

Curved shapes. To handle parts with a non-straight axis, we first simplify the problem by assuming that the general axis lies in a plane. Secondly, we treat the part as being a blend of two straight-axis sub-parts, placed at the two ends of the part. The position of each of these sub-parts is determined by a single depth value in the optimization above, and the whole part is defined by connecting the two subparts with a general axis while constraining the profile snapping.

6 Experimental Results

Our 3-sweep interactive technique has been implemented in C++. The system provides an outline view for 3-sweep interaction, a solid model view, and a texture view for checking the model and image editing. The user can choose between cuboid, cylinder and sphere primitives using a button or key shortcut. The system also provides conventional menu selection, view control and deformation tools. The technique has been tested and evaluated on a large number of photos as we demonstrate in this section and in the accompanying video. As shown in the video, most of the examples were modeled in a few minutes or less. The modeling process is intuitive and fluent, and can be used by unskilled persons following very little training. Editing and repositioning an object requires an effort similar to using other parametric editing techniques.

Once the object has been modeled, we can map the texture from the image onto the object, as shown in Figure 6. By projecting a vertex of the mesh to the image plane, we can get the 2D coordinates of the vertex in the image, which are then used as texture coordinates to map the corresponding part of the image onto the model. As there is no information regarding the back of the object, we simply use a symmetry assumption and mirror the front texture to the back. For each profile layer of the model, we assign the same texture coordinate for the two vertices which are mirrored symmetrically about the center of the layer. Note that on the two sides of the object, there could be centro-symmetric pairs that both face away from the camera. To deal with this situation, we treat the texture associated with these vertices as holes, and use the texture that is present to fill them with a fast image completion technique [Xiao et al. 2011].

Modeling from single image and editing The acquired 3D model and its texture allow semantic image editing. Before editing, the image of the 3D model is cut out from the photo, leaving a black hole which is filled again using an image completion tech-

Figure	1	6					7	8				9		11		
Example	menorah	(a)	(b)	(c)	(d)	(e)	Obelisk	tap	holder	lamp	samovar	pot	telescope	trumpet	handle	horn
Time (s)	80+25	75+15	20	35	30	65+35	20	30+25	45+35	40+50	50+20	15+30	100+30	80	30	60
Constraints	2	4	2	1	1	1	0	2	1	1	1	0	2	1	1	1

Table 1: *Modeling and editing time (in seconds) and the number of manually provided geo-semantic constraints (added or removed) for each example.*

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

Figure 6: *Modeling objects. Top: input photos. Middle: extracted 3D models (blue) are rotated and repositioned. Bottom: modified objects inserted into the same or a new environment, with their textures.*

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

Figure 7: *Modeling the Obelisk in Paris from two photos. (a) The base of the Obelisk is modeled from a closer view which captures more details. (b) The partial 3D model is transported to a more distant view (in which part of the the base is occluded) to complete modeling. (c) A rotated textured Obelisk; the texture of the transported part is blended into the region it occupied. (d) Details of the base are visible in the close-up of the new view.*

Figure 8: *Modeling and replicating parts for image editing. Orange parts are replicated or deformed.*

Figure 9: *Editing a pot and a telescope. The leftmost images are the original photos. Note that different parts have been non-uniformly scaled differently.*

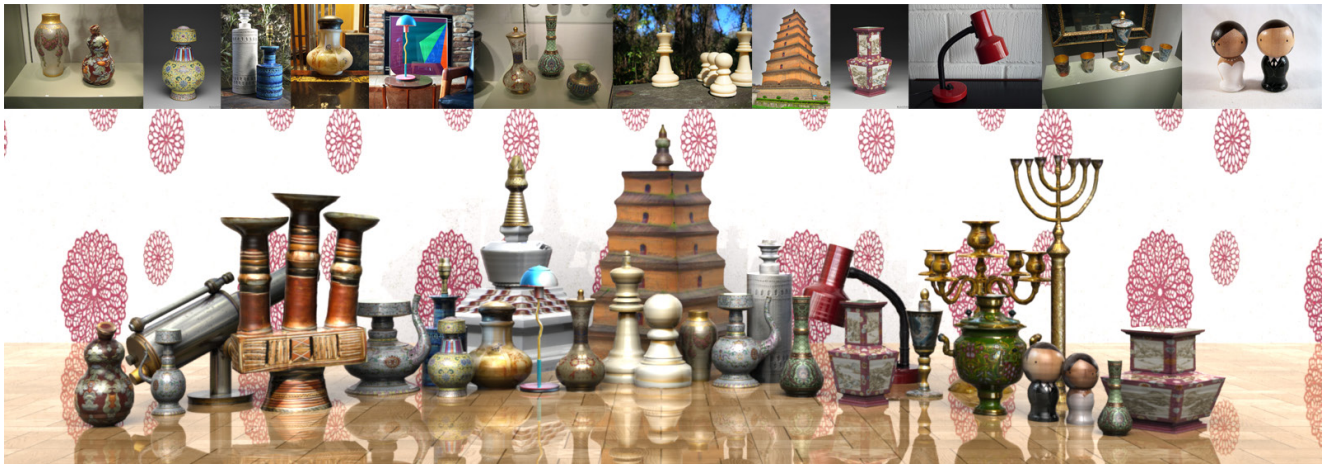


Figure 10: *A rendered scene using source images from the top image strip.*

nique [Xiao et al. 2011].

Figure 1(e) demonstrates a menorah where each arm is rotated through a different angle. All candle holders have the same size, but due to the oblique view, their sizes appear different in the photo. During modeling, we copied each candle holder and fitted each one to the image, while requiring that they lie on the same plane and that their 3D sizes be the same. This efficiently recovered the true 3D position and shape of each part.

Figure 6 shows several modeling results. In the top row we show the input photos, in the middle row we show the extracted and repositioned 3D models, and in the third row, they are inserted with their textures into the same or a new environment. The rightmost column shows the modeling and repositioning of three objects in one complex photo. Note that the menorah has been rotated, and translated on the ground plane.

In Figure 7 we show a case where two input photos are used to model one object: the Obelisk in Paris. Firstly, the base of the Obelisk is modeled from a close up view in (a), allowing more detail to be captured. Then, the partial 3D model is moved to another photo where the entire Obelisk is visible, but the base is occluded. As if performing a copy and paste procedure, the user positions the extracted base inside the image, and it snaps to the image contours in (b). The user then continues the modeling process. The texture of the transported part is blended to match the shading of the region in the new image, to maintain consistency: see the rotated view (c). Details of the base can be seen in the close up view (d) of the final model of the Obelisk.

Figure 8 shows four examples of modeling and editing at part-level, where some parts of the objects (highlighted in gold) are replicated and copied, and optionally rotated to enhance and enrich the shape. At top left is a tap, whose handle is augmented to be 4-sided, and also rotated. The whole tap is also copied and attached to the other side of the wall. The bottom left shows a candle holder modeled and rotated, with its two arms duplicated into a perpendicular position to give four arms. We have also enlarged the middle holder. The top right shows a street lamp with duplicated lamps moved to a lower position and rotated; it has also been copied to other positions in the street. The bottom right shows a samovar, rotated, and additional copies of its handles attached to its surface.

Figure 9 shows various different editing operations carried out on two objects. Note that different scaling has been applied to different object parts. In Figure 10 we show a photograph with a collection of objects that were modeled and copied from other photos.

The supplementary video shows how these objects wered modeled and edited. The modeling and editing time for each example is shown in Table 1, as well as the number of manually provided geo-semantic constraints. Objects in oblique views typically need more manual constraints, most of which note coplanar axes, which are difficult to infer automatically.

Comparison to sketch based modeling. As discussed in Section 2, our method shares some similarities with the one in Shtof et al. [2013], which models objects from sketches. Earlier we discussed the main differences between the methods. Their system is based on sketches rather than photographs, which makes it easier to assume that parts have sufficient bounding curves around them. It relies on labeling, using a drag and drop metaphor for choosing and positioning the primitives before snapping with the sketches. We make a comparison based on their sketch inputs, as their method cannot handle the examples presented in this paper; see Figure 11. We provide a side-by-side modeling session comparison video showing how their sketch labeling and drag and drop snapping steps are significantly less efficient and less intuitive compared to our 3-sweep method. Our modeling time (60s on average) is significantly lower than the time they report for their technique (180s on average).

(a) (b) (c)

Figure 12: *Failures. (a) Due to perspective projection, the table legs cannot be snapped to the image under a parallelism constraint. (b) Due to the assumption of uniformly scaling profiles, the bottom of the toothpaste tube is not flattened. (c) Snapping fails due to an ill-defined edge caused by the shadow cast by the bottle.*

User study. We conducted a user study to evaluate the usability and efficiency of our tool. Eight novice users of our tool and two expert users of commercial software were asked to participate in a modeling task involving 13 different models from images. The modeling time taken was recorded, and the models generated were evaluated by five different evaluators. The statistics gathered show that using our tool is about 20 times faster than commercial tools, while achieving a comparable modeling quality. Indeed, the models generated by our tool are more faithful to the images thanks to edge snapping. However, due to unfamiliarity of novice users with our profile radius constraint functions, our models are generally less smooth. A further benefit is that as our models provide a direct fit to the images, our tool can automatically texture map the model, which done manually may take an artist several hours. More details of the user study can be found in the supplementary material.

Limitations Our work has several limitations. Firstly, many shapes cannot be decomposed into generalized cylinders and cuboids, and so cannot be modeled using our framework: for example, the base of the menorah in Figure 1. It would be desirable to extend the types of primitives which can be modeled using similar principles. Adopting the generalized sweep template described in [Schmidt and Wyvill 2005] can also extend our application range. To preserve the simplicity and smoothness, we sacrificed certain capabilities in our method. For example, our method does not allow non-uniform scale of the 2D profile while sweeping (and therefore cannot model freer natural shapes), and it cannot generate non-planar extrusions. These two cases could be very ambiguous to infer automatically from a single image without adding more complex user controls, which may break the fluency of the method.

Photographs themselves often have some distortions from an ideal perspective projection (see Figure 12(a)), especially if an object is close to the camera or taken with a wide angle lens. In this case, fisheye correction should be applied before modeling, but we currently do not provide it.

Even for a generalized cylinder, the shape can be ambiguous due to the lack of depth information in the image. We assume that the profile of the cylinder is uniformly scaled at each profile and does not rotate around its main axis. This assumption is not always satisfied, as demonstrated in Figure 12(b). We further assume that the main axis of a curved cylinder or cuboid is mainly visible and parallel to the viewing plane. Using the perspective assumption, we can handle a small amount of skew, but not a large one. It is also extremely hard to recover the 3D for parts extruding along the viewing direction, such as the tap at the front of the samovar in Figure 6(d). Our tool requires some manually added or removed geo-semantic constraints when modeling complex object. Adopting consistent

constraints selection described in [Langbein et al. 2004] can reduce the amount of interaction.

Objects that occupy too little of the image, such as the cross in Figure 6(e), are hard to model accurately. Although the user can use a zoomed-in view, modeling accuracy will be poor due to inaccurate edge detection. A similar problem happens when the object has fuzzy edges, as in Figure 12(c). We cannot texture map the back of objects if they do not have a symmetric texture. We also do not support hollow objects (such as the Eiffel Tower). Lastly, our editing assumes a simple illumination model without shadows. Relighting and shadow computations are currently not supported by our system.

7 Conclusion

We have presented an interactive technique which can model 3D man-made objects from a single photograph by combining the cognitive ability of humans with the computational accuracy of computers. The 3-sweep technique is design to allow extracting an editable model from a single image. The range of objects that our technique can support are objects that consist of simple parts, without much occlusion. As we demonstrated, this range is surprising large to be useful for interactive modeling — our tests show that our method can model a large variety of man-made objects in photographs, as well as objects in sketches. The modeled objects can be edited in a semantically meaningful way, both in the original image, or for use in composing new images. In future, we hope to extend the range of primitives, and to allow modeling of the freer shapes of natural objects. We also wish to add symmetry and smoothness constraints on the shapes as in some previous work. 3-Sweep could also be extended to allow modeling from multi-view images or video, without the help of depth data. The applications demonstrated mainly show editing and manipulation of geometry, but the recovered 3D models and surface normals can be used for re-lighting and material editing.

Acknowledgement We thank the anonymous reviewers for their valuable comments. This work was supported by National Basic Research Project of China (2011CB302205), Natural Science Foundation of China (61120106007 and 61103079), National High Technology Research and Development Program of China (2012AA011903), PCSIRT and Tsinghua University Initiative Scientific Research Program, also the Israel Science Foundation (TAU) and the Israel Science Foundation (grant no. 324/11).

References

ANDRE, A., AND SAITO, S. 2011. Single-view sketch based modeling. In *Proceedings of the Eighth Eurographics Symposium on Sketch-Based Interfaces and Modeling*, 133–140.

ANGELIDIS, A., CANIF, M., WYVILL, G., AND KING, S. 2004. Swirling-sweepers: Constant-volume modeling. In *12th Pacific Conference on Computer Graphics and Applications, 2004. PG 2004. Proceedings*, 10–15.

ARBELAEZ, P., MAIRE, M., FOWLKES, C., AND MALIK, J. 2011. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence 33*, 5, 898–916.

ARIKAN, M., SCHWÄRZLER, M., FLÖRY, S., WIMMER, M., AND MAIERHOFER, S. 2013. O-snap: Optimization-based snapping for modeling architecture. *ACM Transactions on Graphics (TOG) 32*, 1, 6.

BARRETT, W., AND CHENEY, A. 2002. Object-based image editing. In *ACM Transactions on Graphics (TOG)*, vol. 21, 777–784.

BENKO, P., KÓS, G., VÁRADY, T., ANDOR, L., AND MARTIN, R. 2002. Constrained fitting in reverse engineering. *Computer Aided Geometric Design 19*, 3, 173–205.

BINFORD, T. O. 1971. Visual perception by computer. In *IEEE conference on Systems and Control*, vol. 261, 262.

BOLLE, R. M., AND VEMURI, B. C. 1991. On three-dimensional surface reconstruction methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence 13*, 1, 1–13.

CHENG, M., ZHANG, F., MITRA, N., HUANG, X., AND HU, S. 2010. Repfinder: finding approximately repeated scene elements for image editing. *ACM Transactions on Graphics (TOG) 29*, 4, 83.

CHENG, M. 2009. Curve structure extraction for cartoon images. In *Proceedings of The 5th Joint Conference on Harmonious Human Machine Environment*, 13–25.

CHOI, B., AND LEE, C. 1990. Sweep surfaces modelling via coordinate transformation and blending. *Computer-Aided Design 22*, 2, 87–96.

DEBEVEC, P. E., TAYLOR, C. J., AND MALIK, J. 1996. Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, 11–20.

EITZ, M., SORKINE, O., AND ALEXA, M. 2007. Sketch based image deformation. In *Proceedings of Vision, Modeling and Visualization (VMV)*, 135–142.

GAL, R., SORKINE, O., MITRA, N., AND COHEN-OR, D. 2009. iwires: an analyze-and-edit approach to shape manipulation. In *ACM Transactions on Graphics (TOG)*, vol. 28, ACM, 33.

GINGOLD, Y., IGARASHI, T., AND ZORIN, D. 2009. Structured annotations for 2d-to-3d modeling. In *ACM Transactions on Graphics (TOG)*, vol. 28, ACM, 148.

GOLDBERG, C., CHEN, T., ZHANG, F., SHAMIR, A., AND HU, S. 2012. Data-driven object manipulation in images. In *Computer Graphics Forum*, vol. 31, 265–274.

JIANG, N., TAN, P., AND CHEONG, L. 2009. Symmetric architecture modeling with a single image. *ACM Transactions on Graphics (TOG) 28*, 5, 113.

LALONDE, J., HOIEM, D., EFROS, A., ROTHER, C., WINN, J., AND CRIMINISI, A. 2007. Photo clip art. In *ACM Transactions on Graphics (TOG)*, vol. 26, 3.

LANGBEIN, F., MARSHALL, A., AND MARTIN, R. 2004. Choosing consistent constraints for beautification of reverse engineered geometric models. *Computer-Aided Design 36*, 3, 261–278.

LI, Y., WU, X., CHRYSATHOU, Y., SHARF, A., COHEN-OR, D., AND MITRA, N. 2011. Globfit: Consistently fitting primitives by discovering global relations. In *ACM Transactions on Graphics (TOG)*, vol. 30, 52.

MERRELL, P., AND MANOCHA, D. 2011. Model synthesis: A general procedural modeling algorithm. *IEEE Transactions on Visualization and Computer Graphics 17*, 6, 715–728.

METAXAS, D. N. 1996. *Physics-based deformable models: applications to computer vision, graphics, and medical imaging*. Kluwer Academic Publishers.

MILLE, J., BONÉ, R., AND COHEN, L. D. 2008. Region-based 2d deformable generalized cylinder for narrow structures segmentation. In *ECCV 2008*. 392–404.

OH, B., CHEN, M., DORSEY, J., AND DURAND, F. 2001. Image-based modeling and photo editing. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 433–442.

OLSEN, L., AND SAMAVATI, F. F. 2010. Image-assisted modeling from sketches. In *Proceedings of Graphics Interface 2010*, Canadian Information Processing Society, 225–232.

OSWALD, M. R., TOPPE, E., AND CREMERS, D. 2012. Fast and globally optimal single view reconstruction of curved objects. In *IEEE CVPR*, 534–541.

RUSSELL, B., AND TORRALBA, A. 2009. Building a database of 3d scenes from user annotations. In *IEEE CVPR*, 2711–2718.

SCHMIDT, R., AND WYVILL, B. 2005. Generalized sweep templates for implicit modeling. In *Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, 187–196.

SCHMIDT, R., WYVILL, B., AND SOUSA, M. C. 2005. Sketch-based modeling with the blob tree. In *ACM SIGGRAPH 2005 Sketches*, 90.

SCHMIDT, R., KHAN, A., SINGH, K., AND KURTENBACH, G. 2009. Analytic drawing of 3d scaffolds. In *ACM Transactions on Graphics (TOG)*, vol. 28, 149.

SEITZ, S., CURLESS, B., DIEBEL, J., SCHARSTEIN, D., AND SZELISKI, R. 2006. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *IEEE CVPR*, vol. 1, 519–528.

SHTOF, A., AGATHOS, A., GINGOLD, Y., SHAMIR, A., AND COHEN-OR, D. 2013. Geosemantic snapping for sketch-based modeling. In *Eurographics*.

SNAVELY, N. 2011. Scene reconstruction and visualization from internet photo collections: A survey. *IPSJ Transactions on Computer Vision and Applications 3*, 0, 44–66.

TAN, P., ZENG, G., WANG, J., KANG, S. B., AND QUAN, L. 2007. Image-based tree modeling. *ACM Transactions on Graphics (TOG) 26*, 3, 87.

TERZOPOULOS, D., WITKIN, A., AND KASS, M. 1988. Constraints on deformable models: Recovering 3d shape and nonrigid motion. *Artificial intelligence 36*, 1, 91–123.

TSANG, S., BALAKRISHNAN, R., SINGH, K., AND RANJAN, A. 2004. A suggestive interface for image guided 3d sketching. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, 591–598.

XIAO, C., LIU, M., YONGWEI, N., AND DONG, Z. 2011. Fast exact nearest patch matching for patch-based image editing and processing. *IEEE Transactions on Visualization and Computer Graphics 17*, 8, 1122–1134.

XU, K., ZHENG, H., ZHANG, H., COHEN-OR, D., LIU, L., AND XIONG, Y. 2011. Photo-inspired model-driven 3d object modeling. In *ACM Transactions on Graphics (TOG)*, vol. 30, 80.

XU, K., ZHANG, H., COHEN-OR, D., AND CHEN, B. 2012. Fit and diverse: Set evolution for inspiring 3d shape galleries. *ACM Transactions on Graphics (TOG) 31*, 4, 57.

XUE, T., LIU, J., AND TANG, X. 2011. Symmetric piecewise planar object reconstruction from a single image. In *IEEE CVPR*, 2577–2584.

ZELEZNIK, R. C., HERNDON, K. P., AND HUGHES, J. F. 1996. Sketch: an interface for sketching 3d scenes. In *ACM SIGGRAPH*, 163–170.

ZHANG, X., GAO, Y., AND CAELLI, T. 2010. Primitive-based 3d structure inference from a single 2d image for insect modeling: Towards an electronic field guide for insect identification. In *International Conference on Control Automation Robotics & Vision (ICARCV)*, 866–871.

ZHENG, Y., FU, H., COHEN-OR, D., AU, O., AND TAI, C. 2011. Component-wise controllers for structure-preserving shape manipulation. In *Computer Graphics Forum*, vol. 30, 563–572.

ZHENG, Y., CHEN, X., CHENG, M., ZHOU, K., HU, S., AND MITRA, N. 2012. Interactive images: cuboid proxies for smart image manipulation. *ACM Transactions on Graphics (TOG) 31*, 4, 99.

ZHOU, S., FU, H., LIU, L., COHEN-OR, D., AND HAN, X. 2010. Parametric reshaping of human bodies in images. *ACM Transactions on Graphics (TOG) 29*, 4, 126.

A Derivation of F

For a straight primitive with reference point $\mathbf{R}$, we denote the three orange points in Figure 5 by $\mathbf{P}_m$, $m \in [1,3]$ (order is unimportant). This gives three equations asseting orthogonality in world coordinates: $\overrightarrow{\mathbf{RP}}_m \cdot \overrightarrow{\mathbf{RP}}_n = 0$, for $(m,n) \in \{(1,2),(2,3),(3,1)\}$. We denote the world coordinates of $\mathbf{P}_m$ by (X_m, Y_m, Z_m), screen coordinates by (x_m, y_m), and depth by z_m. For $\mathbf{R}$, we write (X_r, Y_r, Z_r) etc. We thus have:

$$(X_m - X_r)(X_n - X_r) + (Y_m - Y_r)(Y_n - Y_r) + (Z_m - Z_r)(Z_n - Z_r) = 0.$$

By inverse perspective transformation, this becomes:

$$(\frac{Nx_m}{z_m+v} - \frac{Nx_r}{z_r+v})(\frac{Nx_n}{z_n+v} - \frac{Nx_r}{z_r+v}) + (\frac{Ny_m}{z_m+v} - \frac{Ny_r}{z_r+v})(\frac{Ny_n}{z_n+v} - \frac{Ny_r}{z_r+v}) + (\frac{u}{z_m+v} - \frac{u}{z_r+v})(\frac{u}{z_n+v} - \frac{u}{z_r+v}) = 0,$$

where N, u, v are constant when the perspective parameters are fixed. Since the projection is fixed, x_m, y_m, x_n, y_n are all fixed. The only variables are the z values. To solve these equations, we first replace all z valuess by $\bar{z} = z + v$, By multiplying by $\bar{z}_m \bar{z}_n \bar{z}_r^2$ on both sides, and representing $\bar{z}_m$ by $\bar{z}_n$, we get:

$$\bar{z}_m = \frac{(x_m x_n + y_m y_n + c^2)\bar{z}_r^2 - (x_m x_r + y_m y_r + c^2)\bar{z}_r \bar{z}_n}{(x_n x_r + y_n y_r + c^2)\bar{z}_r - (x_r^2 + y_r^2 + c^2)\bar{z}_n},$$

where $c = v/N$. In this representation, we replace the two unknown $\bar{z}$ values by the third, and solve for the third $\bar{z}$ as a function of $\bar{z}_r$. Let $C_{s,t} = (x_s x_t + y_s y_t + c^2)$, where (s,t) can be $1,2,3$ and r. We then find the representation of $\bar{z}_m$ to be:

$$\bar{z}_m = \pm \frac{C_{r,m}^2 C_{n,l} - C_{r,l} C_{r_m} C_{n,m} - C_{r,n} C_{r_m} C_{l,m} + C_{r,r} C_{l,m} C_{n,m}}{C_{r,r}^2 C_{l,n} - C_{r,r} C_{r,l} C_{r,n}} \bar{z}_r.$$

By symmetry, m, n, l can be any permutation of $1,2,3$. Note that the two solutions exactly match the ambiguity of perspective projection of the primitive. We consider the two solutions and use the one that generates the projection that fits the image edges better. This has the form $\bar{z}_m = a\bar{z}_r$, so z_m is linear in z_r. We can easily compute the world coordinates (X_m, Y_m, Z_m) as a function of z_r by inverse perspective transformation. Since the axis endpoints $\mathbf{C}_{i,j}$ are a linear combination of the $\mathbf{P}_m$, we can also determine each of their coordinates as a function of z_r in the form $b/(a(z_r+v))$, where b, v are determined by the perspective, and a comes from the above derivation.