Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computers & Graphics 29 (2005) 563–575
COMPUTERS & GRAPHICS
www.elsevier.com/locate/cag

A freehand sketching interface for progressive construction of 3D objects ☆

M. Masry*, D. Kang[1], H. Lipson

Sibely School of Mechanical and Aerospace Engineering, Cornell University, Ithaca, NY 14853, USA

Abstract

This paper presents an intuitive, freehand sketching application for Computer Aided Design (CAD) that can reconstruct a 3D object from a single, flat, freehand sketch. A pen is used to draw 2D sketches consisting of straight and curved strokes connected at vertices. The sketches are processed by a reconstruction algorithm that uses the angular distribution of the strokes and their connectivity to determine an orthogonal 3D axis system whose projection correlates with the observed stroke orientations. The axis system is used to determine a plausible depth for each vertex. This approach works well for drawings of objects whose edges predominantly conform to some overall orthogonal axis system. A second, independent optimization procedure is then used to reconstruct each curved stroke in the original sketch, assuming that the curve is planar. New strokes can be attached to the 3D object, or drawn directly onto the object's faces. An implementation of the reconstruction algorithm based on Levenberg–Marquardt optimization allows objects with over 50 strokes to be reconstructed in interactive time.
© 2005 Elsevier Ltd. All rights reserved.

Keywords: 3D sketching; Pen-based computing; Optimization; Human–computer interaction; Computer graphics

1. Introduction

Visual methods of communication are often the simplest and most efficient way of conveying information about the shape, composition and relationships of an object's components. Furthermore, visual information often transcends the limitations imposed by spoken or written languages and is necessary in engineering: a major portion of engineering information is conceived, recorded and transmitted in a visual, nonverbal language [1]. In spite of this, little work has been done to create fast, intuitive sketch-based computer aided design (CAD) interfaces for engineers and designers. Conventional CAD user interfaces are typically cumbersome to use and hamper creative flow.

Freehand sketching, the informal drawing of shapes using freeform lines and curves, has remained one of the most powerful and intuitive tools used at the conceptual design stage. Sketches, in contrast to typical Computer Aided Designs, can quickly and easily be created to convey shape information. Simple paper-based sketching also has many drawbacks: the viewpoint is fixed and cannot be changed in mid drawing; the sketch is passive and cannot be directly simulated or analyzed using computational engineering tools (e.g. structural analysis or kinematic simulation); the sketch is tentative and if a

☆This research has been supported in part by US National Science Foundation (NSF) Grant IIS-0428133 and by a Microsoft University Relations research grant for Tablet Computing.

*Corresponding author. Tel.: +1 607 255 0396.
E-mail addresses: mark.masry@cornell.edu (M. Masry), hod.lipson@cornell.edu (H. Lipson).
[1]D. Kang is now with Tongmyong University of IT. Busan, Korea.

0097-8493/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cag.2005.05.008

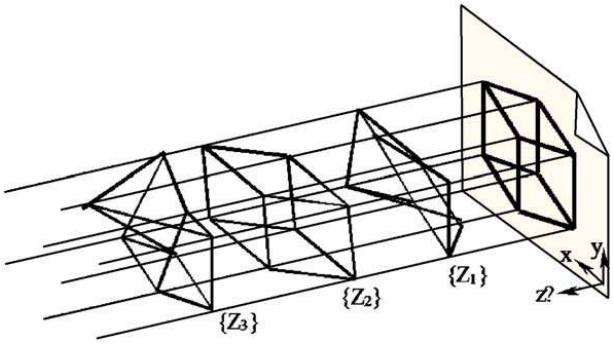


Fig. 1. A user creating, rendering and rotating a shape with a see through hole using the proposed system on a Tablet PC.

final, accurate model is desired, it must be recreated from scratch. The ideal solution from a designer's point of view should combine both the speed and ease of freehand sketching with the flexibility and analytical abilities of CAD tools.

This paper presents an intuitive, pen-based sketching tool that can reconstruct a 3D object from a single, flat, freehand sketch without relying on a database of existing models. As shown in Fig. 1, a user can make an initial sketch, reconstruct it, and add detail using a consistent sketching interface. The proposed system can reconstruct sketches consisting of both straight lines and planar curves. A series of optimization-based reconstruction algorithms are used to achieve this goal. The optimization algorithms run in interactive time on complex sketches, providing a seamless interface for the construction and refinement of 3D objects.

2. Previous work

Systems that use sketch-based input have been the focus of much research. Stahovich et al. [2] demonstrated a system that could interpret the causal functionalities of a 2D mechanism depicted in a sketch, and generate alternative designs. Davis [3] recently showed a system that simulated rigid-body dynamics of a sketched 2D mechanism. These systems are largely 2D.

Fig. 2 outlines the reconstruction of a 3D object from a 2D sketch, in which any arbitrary set of depths $\{Z\}$ that are assigned to the vertices in the sketch constitutes a 3D configuration whose projection will match the given sketch exactly. In principle, each such assignment yields a valid candidate 3D reconstruction. A considerable amount of research has focused on the reconstruction of polyhedral objects from straight-line sketches. Line labeling approaches [4,5] classify each line as convex, concave or occluding edge without explicitly reconstructing 3D shapes. Several methods construct relationships between the slope of sketch lines and the gradients of the associated 3D faces in an attempt to constrain the number of possible interpretations [6,7].

Other methods construct 3D objects incrementally by attaching facets sketched by the user in 2D [8,9]. A gesture-based system for interactively constructing 3D rectilinear models was proposed by Zeleznik et al. [10]. Other approaches to the reconstruction problem require the assumption that the 3D elements in a scene are specified entirely by known primitives [11]. Though restrictive, this allows the reconstructed scene to be specified with a convenient solid geometry.

Optimization-based reconstruction determines the depth assigned to the sketch vertices by optimizing a target function. These methods are more general than the approaches above and can be used to reconstruct relatively complex 3D objects. Optimization-based approaches characterize the relationship of a 2D sketch to an underlying 3D object using systems of linear equations for which the existence of solutions is a sufficient criterion for reconstruction. Linear programming optimization techniques may provide these solutions [12,13]. Another approach is taken by Lipson and Shpitlani [14]: 2D sketches are converted to line and vertex graphs, which are analyzed for regularities such as parallelism, perpendicularity and symmetry. Regularities in the 2D sketch plane are then weighted according to the probability that they correspond to 3D geometrical relationships, and summed to produce an overall compliance function that estimates how well the 3D construction conforms to the regularities in the 2D sketch. Reconstruction proceeds by optimizing this compliance function. There are also statistical approaches to optimization-based reconstruction [15,16]. The correlation between the 2D angles formed by lines in the sketch plane and the angle between these lines in 3D space are learned from a large number of computer-generated 3D shapes and the corresponding projections of these shapes onto a viewing plane. These 2D–3D geometric correlations are then used to determine the most likely 3D shape corresponding to a set of 2D angles, by optimizing over possible assignments of depth values.

While flexible, optimization-based methods suffer because the optimization surface itself may contain many local minima that make it difficult to find the global minimum, while the computational complexity of the optimization process grows rapidly in the number of vertices and lines in a sketch. In contrast to the number of approaches for reconstructing polyhedral objects specified by straight line sketches, there are relatively few reconstruction algorithms that can be applied to sketches of 3D objects with curves. The best-known such work is the Teddy system proposed by Igarashi et al. [17], which uses a sketch-based interface to specify the

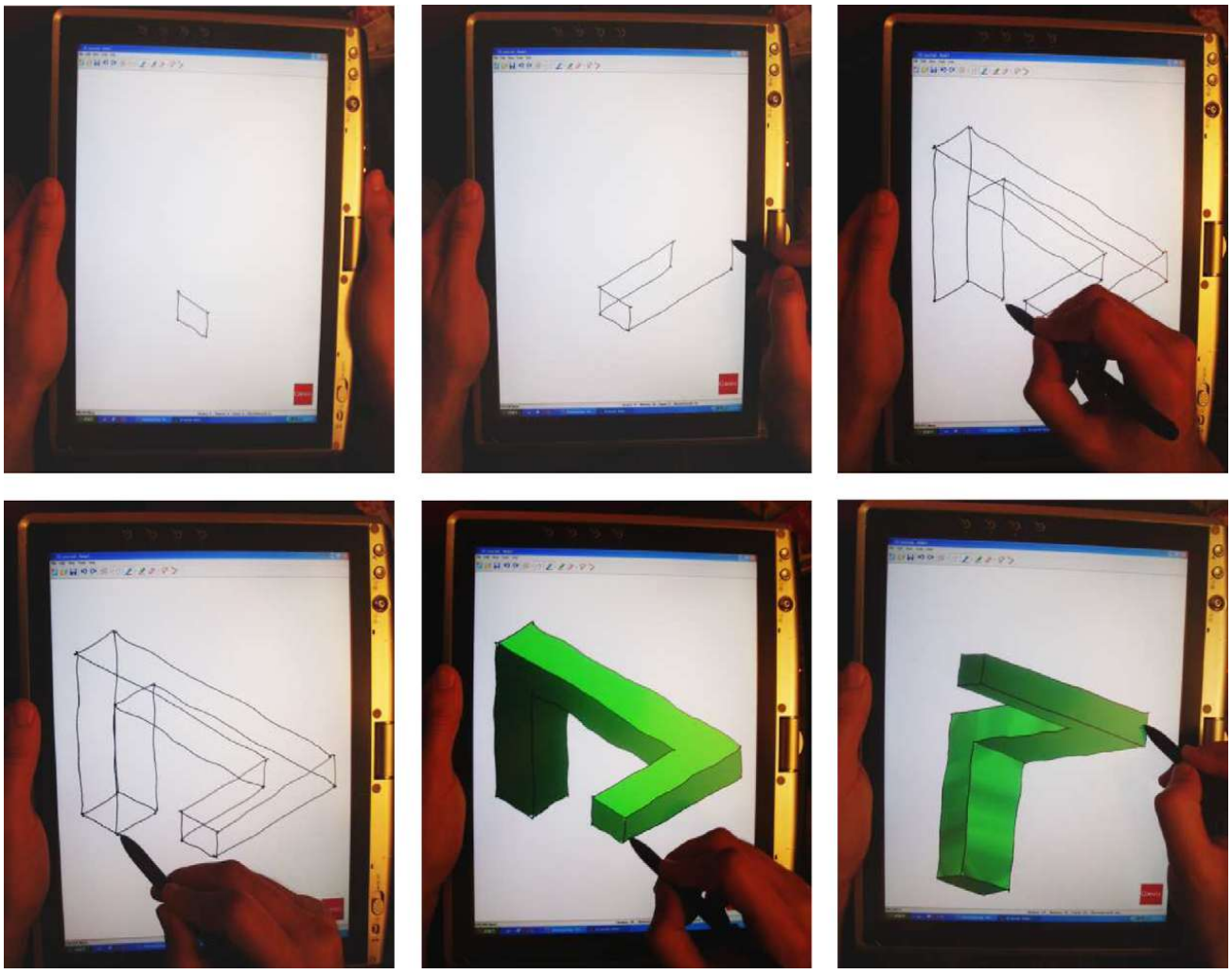


Fig. 2. A sketch provides only two of the coordinates (x, y) of object vertices. A 3D reconstruction must recover the unknown depth coordinate z. In parallel projections, this degree of freedom is perpendicular to the sketch plane; there are an infinite number of candidate objects—the problem is indeterminate. Each candidate object is represented by a unique set of Z coordinates, e.g. sets $\{Z_1\}$, $\{Z_2\}$ and $\{Z_3\}$.

boundaries for reconstruction of a curved solid. This system cannot be used to reconstruct polyhedral objects, or objects that mix straight lines and curves.

The 3D sketching system proposed in this paper uses a fast, optimization-based reconstruction algorithm that chooses a plausible three-connected sketch vertex to serve as a 3D axis origin based on the angular distribution of the lines in a sketch, and reconstructs the depths of the three vertices at the opposite ends of the attached strokes. Depths are then assigned to the other sketch vertices by propagation across the connectivity graph given by the sketch. This approach allows the reconstruction of 3D objects with a connectivity graph whose edges conform to an underlying, orthogonal axis system. Following reconstruction of the sketch vertices, a second optimization procedure reconstructs each curved stroke.

3. 3D sketching system

The sketching system attempts to create an experience similar to drawing with pencil and paper. The application was implemented using the Microsoft Tablet PC API. An example session is shown in Fig. 2. The system allows users to make an initial sketch by drawing strokes using the pen, reconstruct it, and subsequently add new strokes. Following the initial reconstruction, the sketch can be rotated, rescaled or resized. Each stroke is treated as an independent object, and can be erased or modified by the user either pre- or post-reconstruction.

The user interface relies entirely on the pen. A session begins with an initial sketch specified by a set of loosely connected strokes in the sketch plane given by the digitizer surface. Each potentially curved stroke is assumed to be piecewise linear, and is represented

internally by the location of its two endpoints and a series of values specifying the location of each point along the length of the stroke.

Each new stroke is split into smaller strokes if one or more corners are detected using the methods proposed by Shpitlani and Lipson [18]. Strokes may intersect in the sketch plane, but these intersections are not taken to represent intersections in 3D space; at this stage, strokes may be joined only at the endpoints. The reconstruction process is triggered by pressing on the pen's barrel button. As a first step toward reconstruction, all stroke endpoints within a specified distance of one another are connected using an approach given by Shpitlani and Lipson [18]. The 2D sketch can now be interpreted as a connectivity graph (or straight-line graph) representing the 2D orthographic projection of a 3D object onto the plane $z = 0$, with vertices given by the connections between strokes and edges specified by the straight line connections between vertices.

The reconstruction process first determines the 3D position of all sketch vertices, while all curved strokes are treated as straight line connections between vertices, after which all points along each curve are reconstructed. The system then identifies circuits in the connectivity graph and constructs triangulated faces for each circuit. The reconstruction algorithms run in interactive time, allowing for a fluid interaction with the system. The reconstructed shape can be rotated and resized by dragging using the pen.

Strokes can be added, deleted or partially erased at any time. If a new stroke's endpoint is near a reconstructed 3D feature (vertex, stroke or face), its position is automatically interpolated from the 3D object. Strokes sketched directly onto a planar face are automatically reconstructed by interpolating the position of the stroke's points from the face. A stroke deletion or erasure will automatically cause removal of any faces containing the stroke.

The reconstruction algorithm and the methods by which subsequent strokes can be added to the sketch are described in the following sections.

4. Sketch reconstruction

Since the (x, y) coordinates of each vertex are given in the sketch, reconstructing a 3D object requires assigning a z coordinate (also termed the *depth* value) to each vertex, subject to constraints on the characteristics of the resulting 3D object. It is assumed that the sketch vertices are *connected* i.e. that a path can be constructed from each vertex to every other vertex. It is further assumed that none of the vertices or strokes in the sketch completely obscures other elements of the same kind. Though the reconstruction algorithm proposed in this paper requires that at least one vertex be connected to three strokes that represent projections of the 3D axis system, the algorithm can also be adapted to reconstruct an independent 3D axis system that is not directly associated with any vertex in the sketch.

The algorithm is intended to reconstruct 3D objects whose vertices can be connected by a spanning tree consisting of straight lines aligned with one of 3 orthogonal axes. Sketches consisting of connected planar curves can also be reconstructed, provided that the underlying straight line connectivity graph satisfies this requirement. Though these requirements are restrictive, this approach works well for drawings of objects whose edges predominantly conform to some overall orthogonal axis system, which includes a wide range of engineering design drawings. Objects without an underlying rectilinear frame cannot be reconstructed using this method, but this approach can still be to partially reconstruct the object before a more general optimization-based approach [14] is used to complete the reconstruction.

The reconstruction process proceeds as follows:

(1) The distribution of the 2D angles of all straight lines connecting sketch vertices is tested for the presence of three or more significant peaks. If these are found, the sketch is considered to have one or more underlying 3D axis systems, which are then identified.
(2) The identified 3D axis systems are reconstructed from their 2D projections in the sketch plane and used to reconstruct all sketch vertices.
(3) The points along each curved stroke in the original sketch are reconstructed using a separate reconstruction procedure, under the assumption that each stroke is planar.
(4) Connected circuits of approximately coplanar sketch vertices are identified and used to generate the object's faces.

Once the 3D object has been reconstructed, the user can rotate it, add additional strokes, or sketch directly onto the object's faces.

The reconstruction steps are described in more detail below.

4.1. Identifying axis systems

Since orthogonality is the prevailing trend in most engineering drawings, and the easiest to identify, a statistical analysis of the direction of lines in the sketch is performed to determine whether these are consistent with the projections from an underlying orthogonal axis system. The angular distribution graph (ADG) for a set of lines is a discrete histogram of the 2D angles of the lines relative to the sketch plane.

The ADG is constructed by taking each angle to be the mean of a Gaussian distribution with a fixed variance to reduce sensitivity to noise. The resulting sum of Gaussians is then sampled at 1° intervals to yield the discrete ADG. A discrete, rather than continuous, graph is used to facilitate correlation-based similarity measures. Peaks in the ADG show the prevailing sketch angles; the ADGs of most polyhedral 3D objects have clear peaks. The reconstructed object's axis system should thus have a spatial orientation such that it projects onto the sketch plane at angles corresponding to maxima in the ADG. Fig. 3 shows a 2D sketch and the ADG of the associated straight line sketch.

The local ADG of a vertex is the ADG of all lines attached to the vertex. The first step in the reconstruction process is to select a vertex whose local ADG is most similar to the ADG of a representative set of lines. The similarity between any two discrete ADGs is measured using linear correlation. The vertex whose local ADG has the highest correlation with the ADG of the representative set of lines is chosen to be the origin of an axis system. The three lines attached to this vertex represent the projection of the axes onto the sketch plane.

The depth of each vertex is determined by the projection of the connected lines onto the main axis system. Given that the sketch graph is connected, it is possible to construct a spanning tree that connects each vertex to the axis origin. Depth values are then propagated along this tree, beginning at the axis

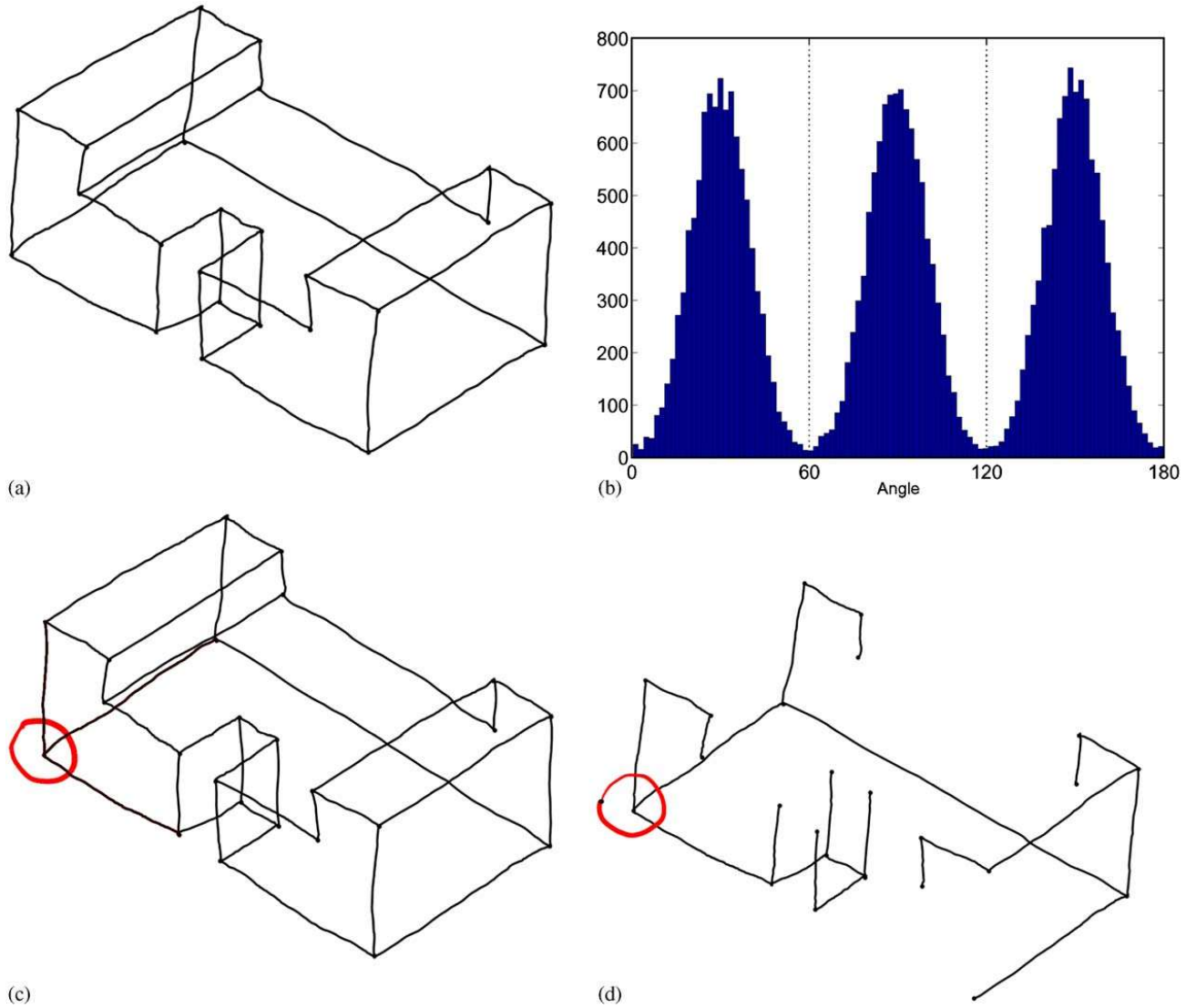


Fig. 3. (a) A 2D sketch with a single distinct axis system, (b) its angular distribution graph (ADG). (c) The sketch with the identified axis vertex circled in red, (d) the sketch's minimum spanning tree (MST) rooted at the selected vertex.

endpoints. The weight assigned to the 2D line direction vector $\mathbf{v}_n = [x_n, y_n]$ associated with line n is given by

$$\max_{\mathbf{v}_a \in \{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}} \frac{|x_n x_a + y_n y_a|}{\sqrt{x_n^2 + y_n^2}\sqrt{x_a^2 + y_a^2}}, \quad (1)$$

where $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}$ is the set of 2D vectors that make up the axis system.

A maximum weight spanning tree (MST) is used to determine the propagation path from the main axis to each vertex. The MST is the tree that connects all sketch vertices such that the sum of the weights of all edges in the tree is maximized. The MST constructed with the weights given by Eq. (1) connects all of the vertices in the sketch while avoiding those lines that are not highly aligned with the main axes; using it to determine the depth values of each vertex therefore minimizes the propagation of reconstruction errors. The MST is determined using Prim's algorithm [19]. The algorithm begins using only the main axis vertex and the projected axes. The tree is then iteratively expanded by selecting the connected line with the highest weight.

The representative ADG and MST for the sketch are determined iteratively in order to minimize the effects of atypical lines using the following algorithm:

(1) Select all lines
(2) Build the ADG of selected lines
(3) Select an axis vertex using the ADG and assign line weights
(4) Generate the sketch MST and select the lines in the MST
(5) If this selection differs from the previous selection, goto step 2

It is assumed that a single MST can be constructed such that all vertices are connected by strokes that are aligned with a single set of 3D axes. If this process does not converge on a single MST after a several iterations, the 2D sketch cannot be reconstructed with the proposed method. An example MST is shown in Fig. 3.

The class of sketches with two distinct axis systems cannot be reconstructed in their entirety with the proposed algorithm. Fig. 4(a) shows a sketch with two axis systems. The sketch ADG shown in Fig. 4(b) has five distinct peaks, as opposed to the three found in the ADGs of sketches with a single axis system such as Fig. 3. If the sketch was drawn using two or more distinct axis systems, the MST construction process will not converge.

4.2. Reconstructing vertex depths

The origin of the main axis system is assumed to have a depth of zero. The depth component of the axis line vectors must be determined in order to reconstruct the main axis system. The x and y components of each axis are given by the vectors $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}$. The unknown z components z_1, z_2, and z_3 are the values that minimize an optimization function based on two assumptions about an ideal sketch:

(1) Since the axis vectors are, ideally, perpendicular to one another in 3D space, the angle between any two axes should be 90° and the cosine of the angle should be 0.
(2) If the length of each line in the sketch plane corresponds to its length in 3D, the difference between the ratio of the axis lengths in the sketch plane and the ratio of their lengths in 3D space should be 0.

Note that the second assumption imposes restrictions on the sketch viewpoint: viewpoints where the orthographic

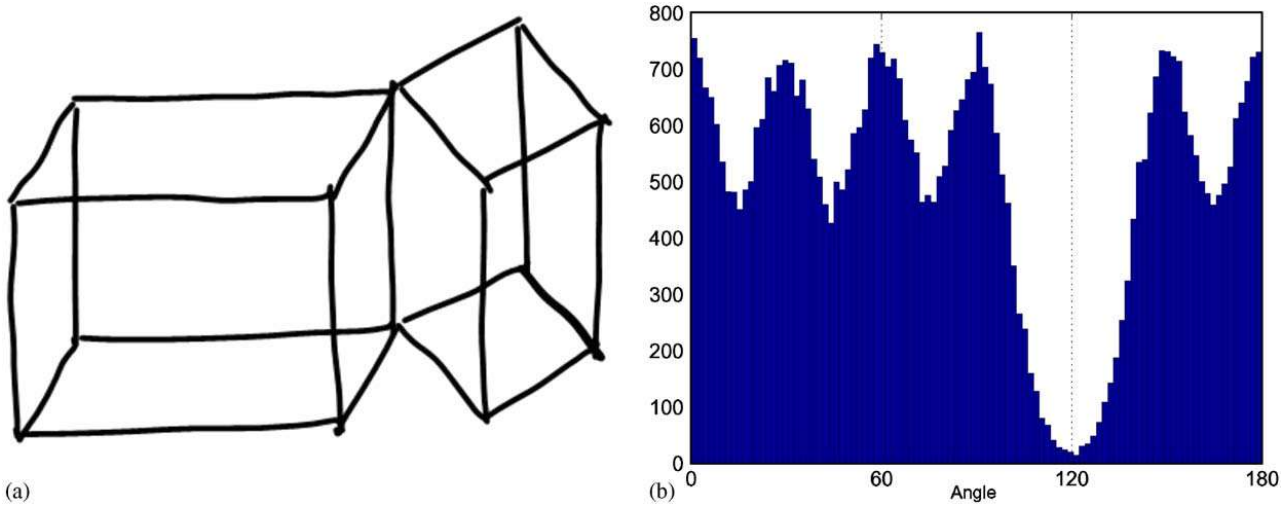


Fig. 4. (a) A 2D sketch with two distinct axis systems and (b) its angular distribution graph (ADG).

projection of the object onto the 2D sketch plane produces axes with very different lengths will result in reconstructed 3D axes with very different lengths.

The optimization goal is therefore to minimize the following cost function $f(z_1, z_2, z_3)$:

$$\begin{aligned} f(z_1,z_2,z_3) &= \cos^2\theta_{21} + \cos^2\theta_{32} + \cos^2\theta_{31} \\ &\quad + \omega\left(\left(r_{21} - \frac{|\mathbf{p}_2|}{|\mathbf{p}_1|}\right)^2 + \left(r_{32} - \frac{|\mathbf{p}_3|}{|\mathbf{p}_2|}\right)^2 \right. \\ &\quad \left. + \left(r_{31} - \frac{|\mathbf{p}_3|}{|\mathbf{p}_1|}\right)^2\right), \\ &= \left(\frac{\mathbf{p}_1\cdot\mathbf{p}_2}{|\mathbf{p}_1||\mathbf{p}_2|}\right)^2 + \left(\frac{\mathbf{p}_3\cdot\mathbf{p}_2}{|\mathbf{p}_3||\mathbf{p}_2|}\right)^2 + \left(\frac{\mathbf{p}_1\cdot\mathbf{p}_3}{|\mathbf{p}_1||\mathbf{p}_3|}\right)^2 \\ &\quad + \omega\left(\left(r_{21} - \frac{|\mathbf{p}_2|}{|\mathbf{p}_1|}\right)^2 + \left(r_{32} - \frac{|\mathbf{p}_3|}{|\mathbf{p}_2|}\right)^2 \right. \\ &\quad \left. + \left(r_{31} - \frac{|\mathbf{p}_3|}{|\mathbf{p}_1|}\right)^2\right), \end{aligned} \tag{2}$$

where $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3$ are the 3D axis vectors, r_{mn} is the ratio of the length of axis lines $\mathbf{p}_m$ and $\mathbf{p}_n$ as measured in the sketch plane, and θ_{mn} is the angle between $\mathbf{p}_m$ and $\mathbf{p}_n$. The weighting factor ω allows a tradeoff between the angular and length constraints. Note that this function has a global minimum at 0.

Since Eq. (2) can be explicitly differentiated, the Levenberg–Marquardt method [20] can be used to determine a solution for this nonlinear optimization problem. This fast nonlinear optimization method is an iterative variation of the Newton method for nonlinear optimization and relies on computation of the Jacobian matrix

$$\mathbf{J} = \left[\frac{\delta f}{\delta z_1}, \frac{\delta f}{\delta z_2}, \frac{\delta f}{\delta z_3}\right]. \tag{3}$$

The partial derivative $\delta f/\delta z_1$, for example, is given by

$$\begin{aligned} \frac{\delta f}{\delta z_1} = &-2\left(\frac{\mathbf{p}_1\cdot\mathbf{p}_2}{|\mathbf{p}_1||\mathbf{p}_2|}\right)\left(\frac{z_2}{|\mathbf{p}_1||\mathbf{p}_2|} + \frac{\mathbf{p}_1\cdot\mathbf{p}_2}{|\mathbf{p}_2|}\right)\left(\frac{z_1}{|\mathbf{p}_1|^3}\right) \\ &-2\left(\frac{\mathbf{p}_3\cdot\mathbf{p}_1}{|\mathbf{p}_3||\mathbf{p}_1|}\right)\left(\frac{z_3}{|\mathbf{p}_3||\mathbf{p}_1|} + \frac{\mathbf{p}_3\cdot\mathbf{p}_1}{|\mathbf{p}_3|}\right)\left(\frac{z_1}{|\mathbf{p}_1|^3}\right) \\ &+2\omega\left(r_{21} - \frac{|\mathbf{p}_2|}{|\mathbf{p}_1|}\right)\left(\frac{z_1|\mathbf{p}_2|}{|\mathbf{p}_1|^3}\right) \\ &+2\omega\left(r_{31} - \frac{|\mathbf{p}_3|}{|\mathbf{p}_1|}\right)\left(\frac{z_1}{|\mathbf{p}_1||\mathbf{p}_2|}\right). \end{aligned} \tag{4}$$

Once the values of z_1, z_2, and z_3 have been determined, the four vertices attached to the axis system are considered to have been reconstructed. The MST can then be used to determine the depths of all other sketch vertices. The depth of any vertex attached to an already reconstructed vertex by a line with direction vector $\mathbf{p}_n = [x_n, y_n, z_n]$ in the MST is determined by first reconstructing the missing depth component of this vector, then adding it to the depth value of the already reconstructed endpoint. The unknown depth component z_n is reconstructed by first selecting the axis vector $\mathbf{p}_a \in \{\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3\}$ that maximizes the 2D projection $(x_n x_a + y_n y_a)/(\sqrt{x_n^2 + y_n^2}\sqrt{x_a^2 + y_a^2})$, then choosing the value of z_n that minimizes the equation

$$\left(1 - \frac{x_a x_n + y_a y_n + z_a z_n}{\sqrt{x_a^2 + y_a^2 + z_a^2}\sqrt{x_n^2 + y_n^2 + z_n^2}}\right)^2. \tag{5}$$

Note that this equation will reach a global minimum of 0 when the direction vector is coincident with a 2D axis vector. This process begins at the axis origin, which has depth 0, and proceeds first to the other vertices connected to the attached axis lines, then throughout the remainder of the MST.

Many alternative methods of constructing the 3D axis system are possible within the formulation presented above. For example, rather than select an axis vertex, an alternative approach is to construct an independent, unattached 3D axis system that will fit the peaks of the ADG. This produces an orthogonal axis system except in degenerate cases, but is more computationally intensive than the ADG algorithm.

4.3. Reconstructing curved strokes

Sections 4.1 and 4.2 dealt with the reconstruction of sketch vertices using a connected straight line graph extracted from the original sketch. Those strokes in the original sketch that cannot be represented by a straight line must also be reconstructed. This is accomplished by a second reconstruction algorithm that processes each stroke independently. The (x, y) locations of every point in a curved stroke are specified in the sketch; the depth of each point in the curve must be determined by the reconstruction process.

Though a stroke can specify an arbitrary path in three dimensions, it is difficult to sketch an arbitrary, unambiguous 3D path entirely by projection onto the sketch plane. The stroke reconstruction algorithm therefore relies on the underlying assumption that each curved stroke is planar, though the parameters of the planar equation are unknown. The goal of the curve reconstruction process is to determine a plane onto which the user might plausibly have drawn the stroke. The depth of each point in the curved stroke is then determined by projection onto the plane (Fig. 5).

The planar equation $a(x - x_0) + b(y - y_0) + c(z - z_0) = 0$ has 3 unknowns $[a, b, c]^T$, which specify the planar normal vector; these must be determined by the reconstruction algorithm. Since the plane is constrained by the requirement that it contain the line $\mathbf{v}$ passing through both of the curve's end points, it is possible to determine the planar normal by optimization over a single variable. An initial planar normal $\mathbf{n}_0 = [a_0, b_0, c_0]^T$

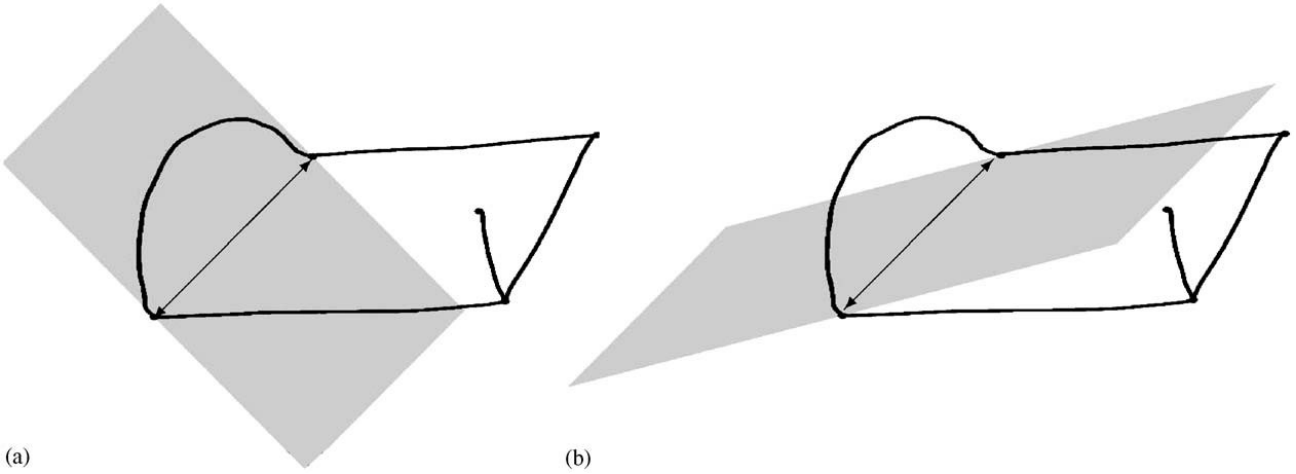


Fig. 5. A 3D axis system with an attached curved stroke, and two possible stroke planes, indicated in light gray. Each projection plane contains the line connecting the two curve endpoints (indicated by the vector). The depth value z_n (where depth is defined into the screen) for each point (x_n, y_n) along the curve are recovered by projection onto the underlying plane. The plane in (a) has the optimal orientation as given by the solution to Eq. (10). The plane in (b) is an implausible plane, which will yield projected depth points well outside the range specified by the two endpoints.

is constructed so that it is perpendicular to $\mathbf{v}$. All other allowable normals can then be constructed by rotating the initial normal by an angle θ around $\mathbf{v}$ to yield the rotated normal $\mathbf{n}_\theta = [a_\theta, b_\theta, c_\theta]^{\mathrm{T}}$:

$$\begin{bmatrix} a_\theta \\ b_\theta \\ c_\theta \end{bmatrix} = [\mathbf{A}_\theta] \begin{bmatrix} a_0 \\ b_0 \\ c_0 \end{bmatrix}, \tag{6}$$

where $\mathbf{A}_\theta$ is a 3×3 rotation matrix that specifies a rotation of angle θ around $\mathbf{v}$. Following rotation of the normal, the equation of the projection plane is given by $a_\theta(x - x_a) + b_\theta(y - y_a) + c_\theta(z - z_a) = 0$, where (x_a, y_a, z_a) is the first stroke endpoint, which is located at one of the reconstructed sketch vertices. The planar equation can likewise be specified by $a_\theta(x - x_b) + b_\theta(y - y_b) + c_\theta(z - z_b) = 0$, where (x_b, y_b, z_b) is the second, similarly reconstructed stroke endpoint.

The optimization function relates the depth value for a particular stroke point to the depths of the stroke's endpoints. The optimization function is based on the assumption that the most likely stroke plane for most simple planar curves is the one that causes the mean depth of the stroke points to fall between the depths of the stroke endpoints:

$$f(n) = (z_n - z_a)^2 + (z_n - z_b)^2. \tag{7}$$

It can be shown that this function has a minimum when $z_n = (z_b + z_a)/2$; minimizing this function over the collection of stroke points will produce mean projected depths that are close to the midpoint of the range between z_b and z_a. Since z_n is determined by projection onto the plane with normal $\mathbf{n}_\theta$ passing through the stroke endpoints (x_a, y_a, z_a) and (x_b, y_b, z_b), Eq. (7) may be rewritten as a function of θ and the stroke points (x_n, y_n)

$$f(\theta, n) = \left(\frac{a_\theta(x_n - x_a) + b_\theta(y_n - y_a)}{c_\theta} \right)^2 \tag{8}$$

$$+ \left(\frac{a_\theta(x_n - x_b) + b_\theta(y_n - y_b)}{c_\theta} \right)^2. \tag{9}$$

The optimization goal for a curved stroke with N points is to find $\theta_{\min}$, where

$$\theta_{\min} = \underset{\theta}{\operatorname{argmin}} \sum_{n=1}^{N} f(\theta, n). \tag{10}$$

The normal of the optimal projection plane is given by $\mathbf{n}_{\theta_{\min}}$.

In many sketches, the projection plane for curved strokes is often related to the other sketch elements. In particular, the normal of the projection plane is often aligned with one of the sketch axes, or with one of the other lines connecting the stroke endpoints. In practice, the optimal projection plane normal is determined by first searching over quantized values of θ to determine a set of normal vectors $\mathbf{n}_\theta$, each of which maximizes the projection onto a single sketch line vector. The optimal normal is chosen from this set as the one that minimizes $\sum_{n=1}^{N} f(\theta, n)$. If the smallest value of $\sum_{n=1}^{N} f(\theta, n)$ measured over this set exceeds a specified threshold, $\theta_{\min}$ is determined by exhaustively searching over a set of quantized angular values.

4.4. Constructing faces

The object's constituent faces must be identified in order to convert the set of 3D vertices and strokes

generated by the reconstruction algorithms into a solid. Several works (e.g. [21]) specify methods of identifying faces in 2D sketches. These algorithms are computationally intensive and relatively complex to implement, however. In this work, faces are identified by recursively searching the connectivity graph of the reconstructed 3D object for cycles of approximately coplanar lines using a fast, greedy search algorithm.

Since each of the object's faces are subtended by a plane, the sketch faces are triangulated by projecting each stroke's points onto the underlying plane, assuming that each descendant face is a hole. Holes are handled by projecting the hole contour onto the underlying plane. The resulting set of 2D contours are then triangulated using a Delauney triangulator capable of handling holes [22]. The resulting triangle mapping is then applied to the 3D points making up each stroke. This approach works well for faces comprised of approximately linear strokes. For faces composed of four strokes, with curved and straight lines appearing in alternate progression, the triangulation is determined using ruled surfaces [23]. More advanced methods (e.g. Coons patches) must be investigated to handle more complex cases.

5. Adding features

The endpoints of all new strokes that are added to the sketch are classified as one of the following:

(1) overlapping an existing reconstructed vertex, in which case this endpoint is linked to the vertex;
(2) lying on a stroke in the 3D object, in which case depth information for the endpoint is determined by interpolating along the 3D stroke;
(3) embedded within a face, in which case the endpoint's depth is determined by projection onto the face. A cycle of strokes whose vertices are all embedded into the same face are said to constitute a descendant face. Faces are stored as a tree; the immediate descendants of a face at the top level are treated as holes.

If none of the above are true, the endpoint's depth is determined using a general but computationally intensive reconstruction algorithm based on the work by Lipson and Shpitlani [19]. The optimization cost function is the sum of separate cost functions given by the parallelism, isometry and orthogonality of the resulting 3D shape:

$$f(\mathbf{v}_m,\mathbf{v}_n) = (1-|\langle \mathbf{v}_m,\mathbf{v}_n\rangle|) + \left(1-\frac{\min(|\mathbf{v}_m|,|\mathbf{v}_n|)}{\max(|\mathbf{v}_m|,|\mathbf{v}_n|)}\right) + (|\langle \mathbf{v}_m,\mathbf{v}_n\rangle|), \quad (11)$$

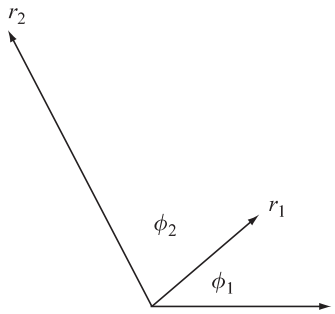
where the vector $\mathbf{v}_n = [x_n, y_n, z_n]$ is the vector given by the difference between the endpoints of stroke n, $\langle \mathbf{v}_m,\mathbf{v}_n\rangle$ is the normalized inner product of $\mathbf{v}_m$ and $\mathbf{v}_n$, and $|\mathbf{v}_n|$ is the vector magnitude. The optimization cost for a sketch consisting of N strokes is given by $\sum_{m=1}^{N}\sum_{n=m}^{N} f(\mathbf{v}_m,\mathbf{v}_n)$. A hill-climber [24] is used to minimize the total cost.

6. Results

The performance of the axis reconstruction algorithm, and the effect of the length ratio weight ω on the reconstructed axis systems, were verified by generating and reconstructing multiple random 2D axis systems. The first axis vector was taken to be the 2D vector $(0,1)$. The second axis was generated by rotating the vector $(0,1)$ by a random angle ϕ_1 and setting its length to a random factor r_1. The third axis vector was generated by rotating the second by a factor ϕ_2 and randomly setting its length to be r_2. The values were chosen such that $0 \leqslant \phi_1 + \phi_2 \leqslant 180^\circ$ and $0.5 \leqslant r_1, r_2 \leqslant 3$. A successful reconstruction generated an assignment of depth values to each axis such that the mean angle $\mu_{\theta_{mn}}$ between the resulting angle satisfied $70 < \mu_{\theta_{mn}} < 110$. Fifty different randomly generated axes were used to evaluate performance for five different values of ω ranging from 1 to 0. The relative importance of length ratios in the optimization function increases as ω increases.

Fig. 6 shows the mean angle between the reconstructed axes (in degrees) and the mean ratio between the longest and shortest reconstructed axes as a function of the weighting term ω, as well as the percentage of random axes that were successfully reconstructed. The percentage of successful reconstructions is encouragingly high, though it drops as ω increases; this is to be expected since the length ratio criterion necessarily limits the number of 3D axis systems that can be successfully reconstructed from a given 2D axis. The mean inter-axis angle $\mu_{\theta_{mn}}$ moves toward the ideal, value of 90° as the effect of axis lengths are reduced, though the mean length ratio μ_{length} increases, resulting in axis lengths that vary considerably. This suggests that the value of ω can be used to determine the elongation of the reconstructed 3D shape.

The curved stroke reconstruction algorithm was tested on several different symmetric and asymmetric curves where the distance $|z_b - z_a|$ between the depths of the two stroke endpoints was varied from 0 to ∞. The algorithm generated plausible reconstructions for curved strokes at different orientations drawn using the same set of fixed endpoints. The reconstruction was most plausible for small to moderate values of $|z_b - z_a|$, and for strokes with a high degree of curvature, though this is to some extent subjective for each user. The stroke planes for strokes with little to no curvature were difficult to determine, largely because the relative



ω	1	0.1	0.01	0.001	0
$\%_{success}$	80%	84%	96%	98%	96%
$\mu\theta_{mn}$	86.98°	89.78°	89.69°	90.18°	90.18°
μ_{length}	1.25	1.24	1.32	1.58	2.09

Fig. 6. A 2D axis system composed of a single fixed axis and two randomly constructed axes with lengths r_1 and r_2 and relative rotation angles ϕ_1 and ϕ_2, respectively. The table shows the percentage of successfully reconstructed 3D axis systems, as well as the mean inter-axis angle $\mu_{\theta_{mn}}$ and mean length ratio μ_{length} measured over the reconstruction of 50 randomly constructed 2D axes.

importance of the orientation of the projection plane decreases as the stroke curves become less pronounced. The reconstruction algorithm was not found to incur significant computational overhead for angular increments of 0.05 rad, and sketches containing up to 10 curved strokes. Since the error term is analytically differentiable, fast optimization techniques may also be employed to reduce computational overhead.

The curved stroke reconstruction algorithm exhibited a strong dependence on viewpoint. In cases where $|z_b - z_a|$ was small, curved strokes were always reconstructed using a projection plane that was nearly parallel to the sketch plane. Furthermore, relatively small variations in the shape of the sketched curve could result in very different reconstructions. Finally, since the curve reconstruction process occurs after the sketch vertices have been reconstructed, the reconstructed vertices may occupy slightly different positions in 3D space than intended by the user, which affects the correspondence of the 2D curve with the intended 3D curve. Further study is required to establish the impact of these considerations on the correspondence between the reconstructed 3D curve and the intended 3D curve.

The reconstruction algorithm performed best on sketches that exhibited strong orthogonal trends, and whose strokes were highly correlated with the underlying axis system, a class of sketches that includes the majority of engineering diagrams. Because the computationally intensive nonlinear optimization is used only to reconstruct the main axis system, rather than depths of all sketch vertices directly, the algorithm can process sketches composed of 50 or more strokes in interactive time on a Pentium 4 Tablet PC notebook computer. An example demonstrating the reconstruction of several sketches incorporating both straight and curved strokes is given in Figs. 7 and 8. The example also demonstrates the addition of holes to the 3D objects, and the performance of the general reconstruction process that optimizes Eq. (11). In all of these examples $\omega = 0.01$.

7. Conclusions and future work

This paper presented a pen-based sketching system for progressively constructing 3D objects from single free-hand sketches. Sketches representing the orthographic projection of a 3D object onto a sketch plane are treated as graphs consisting of vertices connected by the sketch strokes. The reconstruction problem consists of assigning a depth value to each vertex, and subsequently reconstructing each curved stroke. This is accomplished by a two stage reconstruction process. The first stage tests a straight line sketch extracted from the original for the presence of prevailing angular trends and uses these to determine a 3D axis system that maps 2D lines onto 3D lines. This axis system is used in combination with the sketch connectivity graph to assign a depth to each vertex. The second stage reconstructs curved strokes, under the assumption that they are planar. Following reconstruction, the 3D object's faces are identified and triangulated. Users can then add additional strokes, and sketch directly onto the object's faces. The system is implemented with a consistent pen-based interface that mimics pencil and paper sketching, and can reconstruct sketches of up to 50 strokes in interactive time.

Future work on the ADG reconstruction algorithm will address the reconstruction of shapes that do not exhibit pronounced angular trends, or that have strokes with large deviations from the underlying axis system. Future work on the curve reconstruction algorithm will address the problem of jointly reconstructing multiple curves in order to increase the symmetry of the 3D object. Furthermore, examination of the optimization surface for several curved strokes showed a plateau around the global minimum, suggesting that there are a range of possible stroke planes that might yield very similar results. Adding a term that maximizes the projection of the planar normal onto the sketch axis system to the optimization function might differentiate between these planes, and allow the overall angular trends in the sketch to be incorporated into the curve

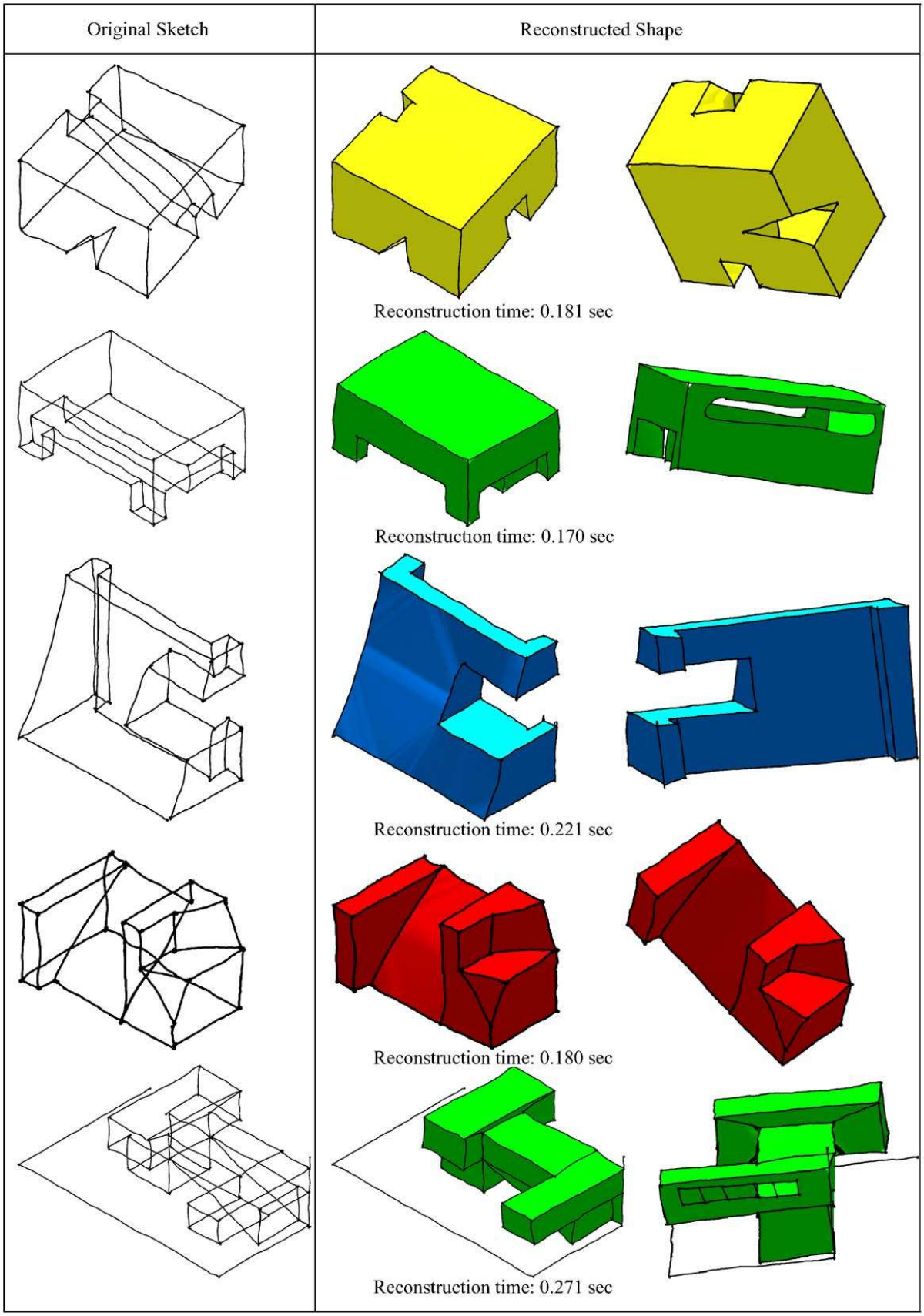


Fig. 7. (a) Symmetrical unreconstructed straight-line 2D sketches and the accompanying reconstructed 3D shapes from two viewpoints. Reconstruction times are given for a Pentium 4M 1.7 Ghz Tablet PC.

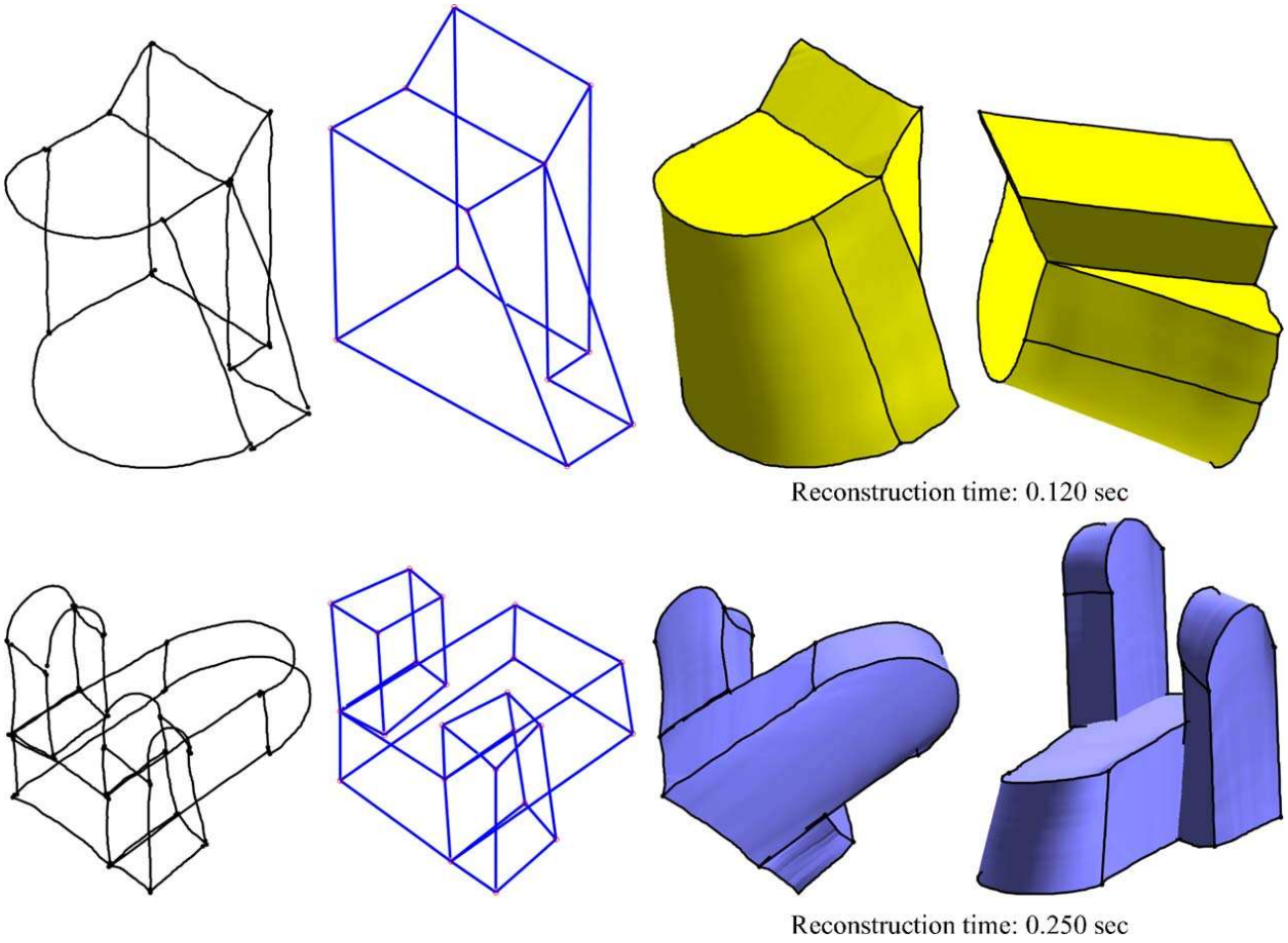


Fig. 8. Two sketches of shapes with mixed curved and straight lines, the underlying straight-line graphs used to reconstruct vertex depths, the reconstructed objects, an alternate viewpoint.

reconstruction process. Finally, research on the construction of plausible 3D surfaces from object faces consisting of multiple curved planar strokes is also required.

References

[1] Ferguson ES. Engineering and the minds eye. Cambridge, MA: MIT Press; 1992.
[2] Stahovich TF, Davis R, Schrobe J. Generating multiple new designs from a sketch. Artificial Intelligence 1998;104: 211–64.
[3] Davis R. Position statement and overview: sketch recognition at MIT. In: AAAI Sketch Understanding Symposium. Stanford, CA; 2002.
[4] Huffman DA. Impossible objects as nonsense sentences. In: Meltzer B, Mitchie D, editors. Machine intelligence. Edinburgh: Edinburgh University Press; 1971.
[5] Clowes MB. On seeing things. Artificial Intelligence 1971;2(1):79–112.
[6] Mackworth AK. Interpreting pictures of polyhedral scenes. Artificial Intelligence 1973;4:121–37.
[7] Wei X. Computer vision method for 3D quantitative reconstruction from a single line drawing. PhD thesis, Department of Mathematics, Beijing University, China [in Chinese; for a review in English see Wang and Grinstein; 1993].
[8] Lamb D, Bandopdhay A. Interpreting a 3D object from a rough 2D line drawing. In: Proceedings of visualization '90; 1990.
[9] Fukui Y. Input method of boundary solid by sketching. Computer aided design 1998;20(8):434–40.
[10] Zeleznik R, Herndon K, Hughes J. SKETCH: an interface for sketching 3d scenes. In: Proceedings of SIGGRAPH; 1996.
[11] Wang W, Grinstein G. A polyhedral object's CSG-Rep construction from a single 2D line drawing. In: Proceedings of the 1989 SPIE intelligent robots and computer vision III: algorithms and techniques, vol. 1192; 1989.
[12] Sugihara K. The interpretation of line drawings. Cambridge, MA: MIT Press; 1986.
[13] Grimstead IJ, Martin RR. Creating solid models from single 2D sketches. In: Solid modeling '95. Salt Lake City, UT, USA; 1995.

[14] Lipson H, Shpitlani M. Optimization-based reconstruction of a 3D object from a single freehand line drawing. Journal of Computer Aided Design 1996;28(8):651–63.
[15] Lipson H, Shpitlani M. Conceptual design and analysis by sketching. Journal of AI in Design and Manufacturing (AIEDEM) 2000;14:391–401.
[16] Lipson H, Shpitlani M. Correlation-based reconstruction of a 3D object from a single freehand sketch. In: AAAI spring symposium on sketch understanding; 2002.
[17] Igarashi T, Matsuoka S, Tanaka H. Teddy: sketching interface for 3d freeform design. In: Proceedings of SIGGRAPH; August 1999.
[18] Shpitlani M, Lipson H. Classification of sketch strokes and corner detection using conic sections and adaptive clustering. ASME Journal of Mechanical Design 1997;119(2): 131–5.
[19] Cormen TH, Leiserson CE, Rivest RL. Introduction to algorithms. Cambridge, MA: MIT Press; 1999.
[20] Heath MT. Scientific computing, an introductory survey, 2nd ed. New York, NY: McGraw-Hill; 2002.
[21] Shpitlani M, Lipson H. Identification of faces in a 2D line drawing projection of a wireframe object. IEEE Transactions on Pattern Analysis and Machine Intelligence 1996;18(10):1000–12.
[22] Shewchuk JR. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In: Lin MC, Manocha D, editors. Applied computational geometry: towards geometric engineering, Lecture Notes in Computer Science, vol. 1148, Springer, Berlin; p. 203–22, from the First ACM Workshop on Applied Computational Geometry.
[23] Schneider PJ, Eberly DH. Tools for computer graphics, 2003.
[24] Press WH, Teukolsky SA, Vetterling WT, Flannery BP. Numerical Recipes in C++: The art of scientific computing, 2nd ed. London, New York: Cambridge University Press; 2003.