EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling (2005)
Takeo Igarashi, Joaquim A. Jorge (Editors)

# A Sketch-Based Interface for Iterative Design and Analysis of 3D Objects

M. Masry and H. Lipson

mark.masry,hod.lipson@cornell.edu
Sibely School of Mechanical and Aerospace Engineering
Cornell University

**Abstract**

*This paper presents a freehand, sketch-based interface for Computed Aided Design (CAD) engineering design and finite element analysis. After a user sketches a two dimensional sketch consisting of connected straight and curved strokes, the sketch is processed by two optimization-based reconstruction algorithms that can reconstruct sketches of 3D objects made up of straight lines and planar curves. The proposed implementation allows certain types of objects with over 50 strokes to be reconstructed in interactive time. Following reconstruction, the structural properties of the 3D shape can be examined using finite element analysis. The object can quickly be modified using the pen-based interface according to the results of the analysis. The combination of a rapid, sketch-based design interface and finite element analysis allows users to iteratively design, analyze and modify 3D objects in an intuitive and flexible way.*

Categories and Subject Descriptors (according to ACM CCS): I.3.6 [Computer Graphics]: Interaction Techniques

## 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. Visual information often transcends the limitations imposed by spoken or written languages. Engineering information, in particular, is often conceived, recorded and transmitted in a visual, non-verbal language. Little work has been done, however, to create fast and intuitive conceptual interfaces based on visual input. 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 CAD 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 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 computational analysis tools. Sketch reconstruction algorithms allow designers to quickly specify a 3D object using a single, freehand sketch; the object can then be subjected to real-time physical simulations such as structural, fluid, or manufacturing analysis. The combination of sketching and physical simulation allows for a revolutionary, *iterative* design process: users can sketch an object, gain immediate insight into its physical properties, and revise the sketch until the design concept matures. By eliminating the restrictions of traditional CAD interfaces, analysis tools can be brought much earlier into the critical early conceptual design stage.

In addition to its practical applications, an interface for 3D sketching and analysis has significant educational benefits. By removing the barrier between sketch and simulation students can quickly explore and understand the physical properties, the advantages and the weaknesses of their design. Instructors can quickly convey design concepts and physical

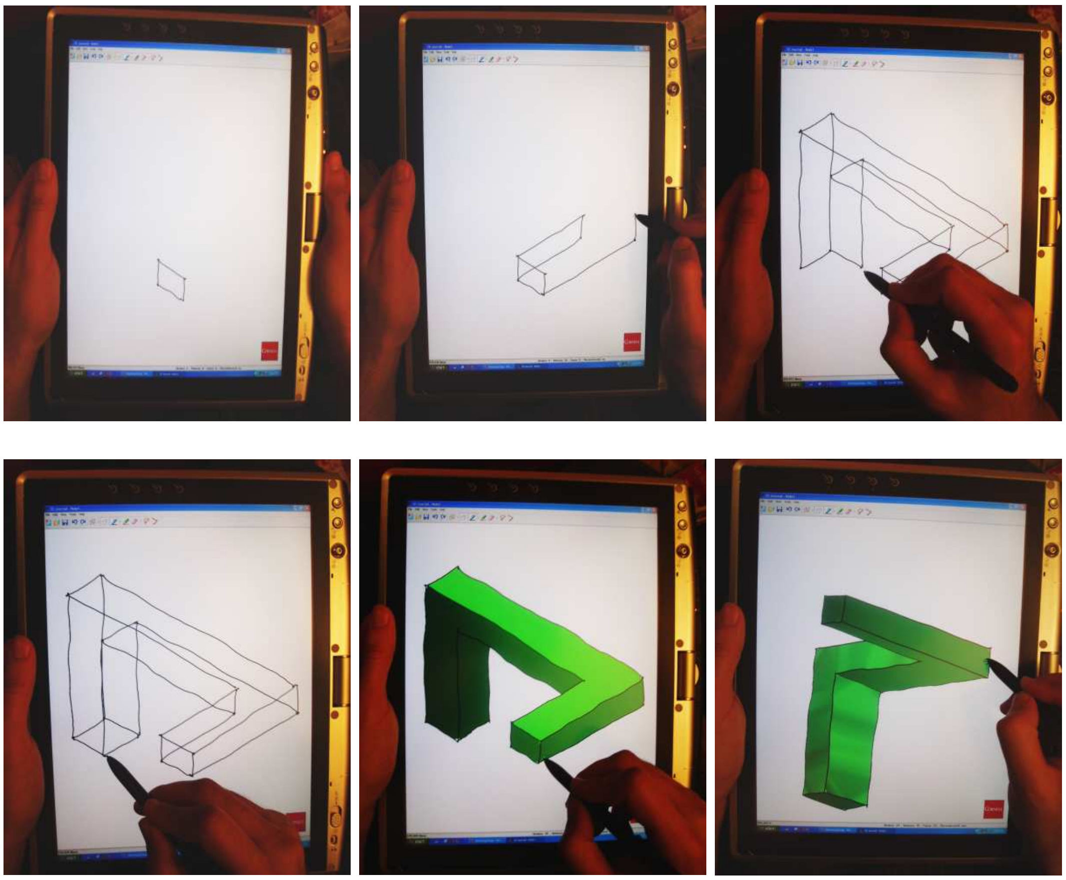


**Figure 1:** *A user creating, rendering and rotating a shape using the proposed system on a Tablet PC*

properties during lectures, making the classroom experience more dynamic and facilitating learning.

This paper presents an intuitive, pen-based sketching tool that combines sketch-based design of 3D objects with finite element analysis (FEA) in order to achieve these goals. The proposed approach consists of two parts: a reconstruction stage that reconstructs a 3D object from a single orthographic sketch, and an analysis stage that performs a finite element analysis on the resulting 3D object and displays the results in the sketch plane. The reconstruction algorithms can process sketches consisting of both straight lines and planar curves, and run in interactive time on certain types of complex sketches. Once the analysis has been performed, users can modify the shape using a consistent pen-based interface, and perform subsequent analyses until the design is complete.

## 2. Previous work

Several works have investigated the use of sketch-based interfaces. Stahovich et al. [SDS98] demonstrated a system that could interpret the causal functionalities of a two dimensional mechanism depicted in a sketch, and generate alternative designs. Davis et al. [Dav02] recently showed a system that simulated rigid-body dynamics of a sketched two-dimensional mechanism. These systems are mostly two dimensional, and the few that are 3D require additional steps that break the flow of sketching.

Figure 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. Several methods construct relationships between the slope of sketch lines and the gradients of the associated 3D faces [Mac73, Wei87]. Other approaches include incremental construction [LB90, Fuk98, ZHH96], and construction using known primitives [WG89]. Detailed surveys are given in [LS00] and [CPC04].

Optimization-based reconstruction for polyhedral objects

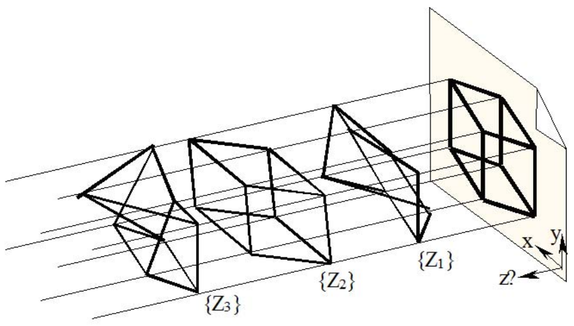


**Figure 2:** *A sketch provides only two of the coordinates $(x, y)$ of object vertices. A 3D reconstruction must recover the unknown depth coordinate z. In parallel projections, this degree of freedom is perpendicular to the sketch plane; there are an infinite number of candidate objects – the problem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

specified by straight line sketches determine the depth assigned to each sketch vertex by optimizing a target function. These methods are more general than the approaches above and can be used to reconstruct relatively complex 3D objects, though they suffer from high computational complexity and susceptibility local minima in the target function. A comprehensive review of optimization-based reconstruction techniques can be found in [LS96, LS00]. We proposed an approach in which 2D sketches were converted to line and vertex graphs and analyzed for regularities such as parallelism, perpendicularity and symmetry [LS96]. These regularities were then summed to produce an overall compliance function that measures how well the 3D construction conforms to the regularities in the 2D sketch. Reconstruction proceeds by optimizing this compliance function. We have also investigated machine learning approaches to sketch reconstruction [LS02]. Optimization-based approaches to 3D reconstruction were used by Shesh et al. [SC04] in conjunction with incremental shape construction methods. Optimization-based approaches to sketch beautification and reconstruction were also used by Company et al. [CCCP04]. Neither of these two methods, however, address the problem of curve reconstruction, or incorporate physical analysis into the design process.

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 [IMT99], which uses a sketch-based interface to specify the boundaries for reconstruction of a curved solid. This system cannot be used to reconstruct polyhedral objects, or objects that mix straight lines and curves. Approaches that deform a template (e.g. [VTMS04]) to fit a curved structure have also been developed, though these require that a 3D template be formed prior to curve reconstruction, and can therefore not reconstruct arbitrary single curves.

The 3D sketching system proposed in this paper uses a fast 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.

In this work we wish to explore the utility of 3D sketch understanding as a design tool, when combined with conventional analysis capabilities. We chose to link the sketch interpretation with standard finite element analysis code [RP05], and demonstrate a complete cycle of 3D modeling and result presentation performed entirely in the sketch medium. We chose Finite element analysis as a commonly used form of engineering analysis with a broad spectrum of applications, though clearly other analysis codes could be used.

## 3. 3D Sketching System

This sketching system attempts to create an experience similar to drawing with pencil and paper. The application was implemented on a tablet PC with a pen input device. An example session is shown in Figure 1. 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. 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. Users can erase all or part of a stroke at any time.

The reconstruction process when a user attempts to spin the 3D object 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 et al. [SL97]. 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.

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. Following reconstruction, the 3D object's faces are identified and triangulated. The triangulated faces are used to facilitate user interaction with the object, and as part of a hidden line removal algorithm that determines the visible parts of each stroke. The triangulated faces are also used to construct the object manifold surface that is required for finite element analysis.

The 3D strokes making up the reconstructed object may be altered and erased in the same way as the 2D strokes in the original sketch. Strokes may also be added to the 3D object, in which case the reconstruction procedure is used to determine the 3D position of any stroke vertices that are not coincident with object features.

The same interface can be used to specify the face parameters required to perform Finite Element Analysis (FEA). The analysis itself is triggered using the application's toolbar, at which point a tetrahedral mesh is generated for the reconstructed solid and finite element analysis is performed. The resulting deformation is superimposed directly over the strokes making up the original sketch.

The following sections describe the reconstruction and analysis stages in more detail.

## 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. This restriction necessarily restricts the reconstruction algorithm to sketches of single objects. Further research is required to determine the best method by which sketches of multiple objects can be reconstructed with suitable relative positions. It is further assumed that none of the vertices or strokes in the sketch completely obscure other elements of the same kind.

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. The sketch vertices need not necessarily be trihedral (see objects 1 and 4 in Figure 4), though sketches with vertices connected to the spanning tree. In these cases, the proposed approach can be used to reconstruct part of the object before a more general optimization-based algorithm (e.g. [LS96]) is used to complete the reconstruction.

Reconstruction proceeds in two stages: the depths of the sketch vertices are determined while treating all curved strokes as straight line connections between vertices, following which all points along each curved stroke are reconstructed, assuming that the resulting 3D curve is planar. The reconstruction algorithms run in interactive time, allowing for a fluent interaction with the system.

### 4.1. Reconstructing vertex depths

The vertex reconstruction algorithm used in this paper was given by the authors in an earlier work [KML04], and is summarized here for convenience.

Since orthogonality is the prevailing trend in most engineering drawings, and the easiest to identify, a statistical analysis of the direction of the lines connecting the sketch vertices is performed to determine whether these are consistent with the projections from an underlying orthogonal axis system. The reconstruction process begins with the selection of the three-connected vertex whose attached lines are most representative of the angular distribution of a representative set of sketch lines; this vertex is the origin of the main sketch axis system, and the attached strokes are taken represent the orthographic projection of the 3D axes onto the 2D sketch plane.

The origin of the main axis system is assumed to have a depth of zero. The depth of the opposite vertex of each of the three attached axis line vectors must be determined in order to reconstruct the axis system. The unknown depths are determined as the minimizing solution of an optimization function based on two assumptions about an ideal sketch:

1. The 3D axis vectors should be as close to mutually orthogonal as possible.
2. The ratio of the axis lengths in the sketch plane is equal to the ratio of their lengths in 3D space.

Note that the second assumption imposes restrictions on the sketch viewpoint: viewpoints where the orthographic 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 to minimize a cost function that attains a value of 0 when all three axes are orthogonal in 3D and the ratio of the axis lengths in 3D is equal to their ratio in the sketch plane. Minimization of this cost function can be performed with a fast Levenberg-Marquardt algorithm. Since optimization is required only to reconstruct the three vertices of the axis system, it does not depend on the sketch allowing complicated sketch graphs to be reconstructed in real-time.

Since the sketch graph is connected, it is possible to construct a spanning tree that connects each vertex to the origin vertex. The spanning tree is given by the Maximum weight

Spanning Tree (MST), where the weight of each edge is given by the projection of the edge line onto the 2D axis lines. Once the axis system vertices have been reconstructed, the depths of the remaining vertices are determined by propagating depth values along this tree, beginning with the axis origin.

The depth of the endpoint vertices of all strokes added to the sketch post-reconstruction are determined by interpolating from an underlying, reconstructed sketch feature. If the endpoint does not lie over an existing feature, reconstruction is performed using a general, axis-independent but computationally intensive reconstruction algorithm based our earlier work [LS96]. A hill-climber [PTVF03] is used to minimize the total cost. This approach is also used for the initial reconstruction if there exists no vertex that can serve as the origin of the main axis system.

The curve reconstruction algorithm only requires that the vertices be reconstructed and planar faces be identified in order to run; there are several alternative approaches for reconstructing vertex depths. Company et. al [CCCP04] proposed an optimization-based reconstruction engine that also makes use of a minimum spanning tree, as well as a comprehensive set of observed shape regularities and two separate inflation methods. Their approach also includes an optimization-based sketch beautification strategy that may add to computational cost. A wide range of shapes can be reconstructed, although these occasionally require the use of sketch regularities such as planarity and corner orthogonality that are not considered in this work. Varley et al. [VMS04] proposed a framework for labeling the lines in a sketch that begins by analyzing the angular distribution of lines in the sketch, then constructing a 3D axis system by solving a system of linear equations. The reconstructed axis system is used along with line parallelism to determine the depths of all sketch vertices prior to assigning line labels. This methods is also capable of reconstructing a wide range of sketches, though these appear to also conform to an orthogonal axis system. Furthermore, the method was developed for natural sketches, rather than the wireframes under consideration here.

## 4.2. Reconstructing Curves

The $(x,y)$ locations of every point in a curved stroke are specified in the sketch; once the vertex depths have been reconstructed, a second reconstruction algorithm reconstructs the depths of each stroke point. 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. All curves are treated as piecewise linear collections of points in the sketch plane. Special curves (e.g. conic sections) are not treated independently.

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 **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 $\mathbf{n}_0 = [a_0, b_0, c_0]^T$ is constructed so that it is perpendicular to **v**. All other allowable normals can then be constructed by rotating the initial normal by an angle $\theta$ around **v** to yield the rotated normal $\mathbf{n}_\theta = [a_\theta, b_\theta, c_\theta]^T$:

$$\begin{bmatrix} a_\theta \\ b_\theta \\ c_\theta \end{bmatrix} = [A_\theta] \begin{bmatrix} a_0 \\ b_0 \\ c_0 \end{bmatrix} \tag{1}$$

where $\mathbf{A}_\theta$ is a $3\times3$ rotation matrix that specifies a rotation of angle $\theta$ around **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 users intended the depth of the stroke points to fall between the depths of the two stroke endpoints. We therefore choose the stroke plane as the one that causes the depth of each stroke points to fall between the depths of the stroke endpoints, which can be satisfied by minimizing the sum squared distance between both endpoints:

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

It can be shown that this function has a minimum when $z_n = \frac{z_b+z_a}{2}$; 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 $(x_b,y_b,z_b)$, Equation 2 may be rewritten as a function of $\theta$ and the stroke points $(x_n,y_n)$

$$f(\theta,n) = \left(\frac{a_\theta(x_n-x_a)+b_\theta(y_n-y_a)}{c_\theta}\right)^2 \tag{3}$$

$$+\left(\frac{a_\theta(x_n-x_b)+b_\theta(y_n-y_b)}{c_\theta}\right)^2 \tag{4}$$

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

$$\theta_{min} = arg\min_{\theta} \sum_{n=1}^{N} f(\theta,n) \tag{5}$$

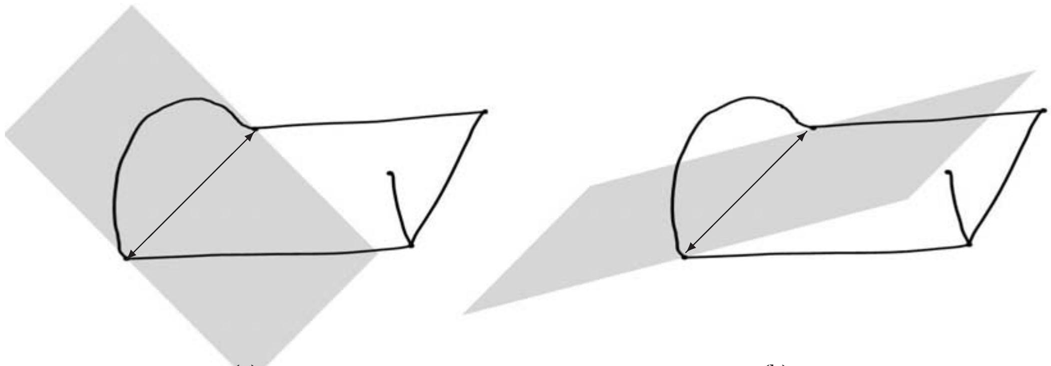


**Figure 3:** *A 3D axis system with an attached curved stroke, and two possible stroke planes, indicated in light gray. Each projection plane contains the line connecting the two curve endpoints (indicated by the vector). The depth value $z_n$ (where depth is defined into the screen) for each point $(x_n, y_n)$ 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 Equation 5. The plane in (b) is an implausible plane, which will yield projected depth points well outside the range specified by the two endpoints.*

The normal of the optimal projection plane is given by $\mathbf{n}_{\theta_{min}}$. A gradient steepest descent algorithm can be used to find $\theta_{min}$. 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, with one of the other lines connecting the stroke endpoints, or with a the normal of a planar face plane in the reconstructed straight line object. 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 vector. The optimal normal is chosen from this set as the one that minimizes $\sum_{n=1}^{N} f(\theta, n)$.

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 $\infty$. 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 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 radians, 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.

## 5. Analysis

Once the 3D object has been reconstructed in its entirety, the object's constituent faces must be identified in order to generate a solid suitable for finite element analysis. 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. The sketch faces are then triangulated. If the face is approximately planar, a Delaunay triangulator [She96] is used. If the face is composed of 3 or 4 non-coplanar curves, the face is triangulated using Coons patches. Faces composed of more than 4 non-planar curved strokes are not triangulated. Note that the use of Coons patches to triangulate curved surface necessarily limits the interpretation of the curved surface. The correspondence between the reconstructed surface and the user-intended surface requires further investigation.

Once the faces in the object have been triangulated, the object's properties can be analyzed and, if necessary modified by the addition of further object features. Though there many different types of analysis can be applied to the 3D shape (e.g. structural, fluid, or manufacturing analysis) we have implemented a linear approximation to the laws for elastic deformation, since this is representative of engineering design analysis.

Finite Element Analysis (FEA) decomposes a continuous function over a global domain into a piecewise series of functions applied to a set of smaller elements whose union represents the original domain. Prior to performing FEA, the reconstructed 3D object is first checked to see if its triangulated faces can be combined to form a triangulated manifold surface. If so, the surface is decomposed into a set of tetrahedral elements using a meshing algorithm [Si05]. The mesh complexity is limited only by the requirements of the mesh-

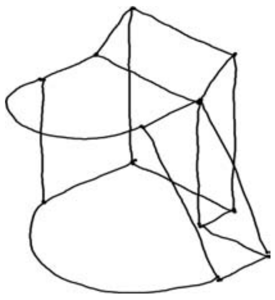
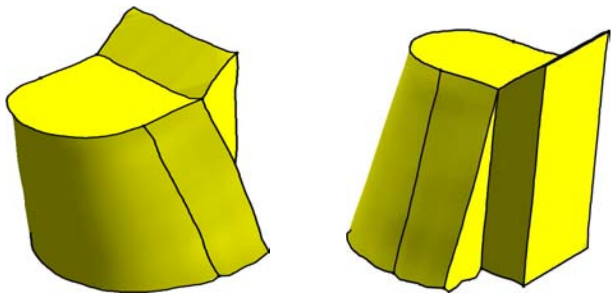
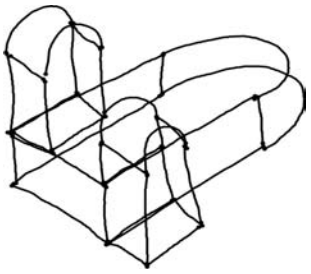
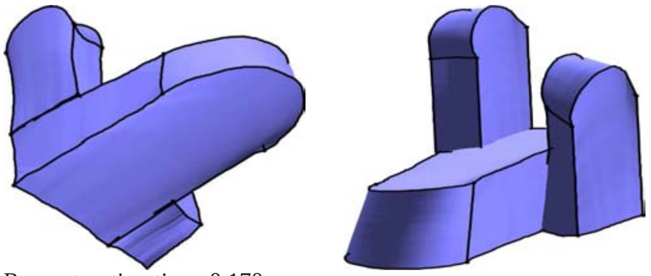
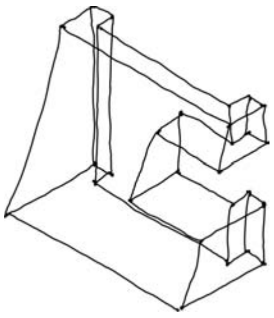
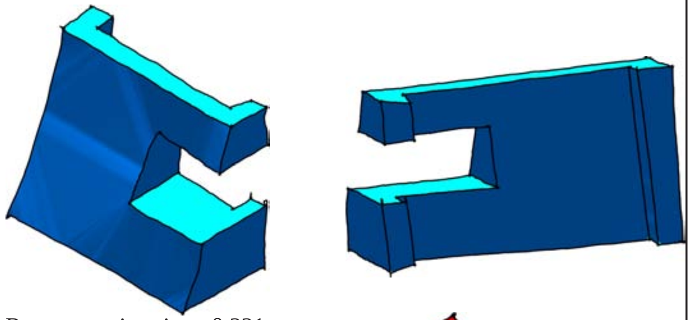
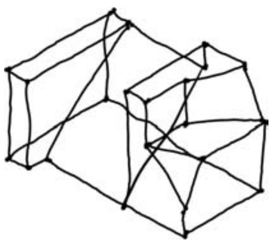
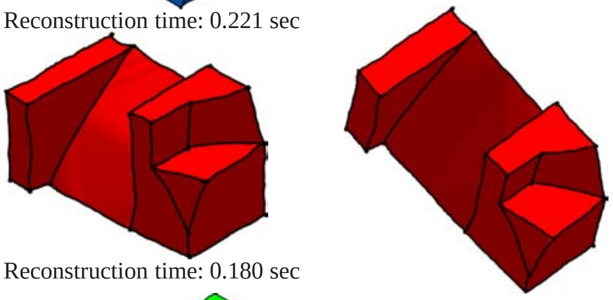
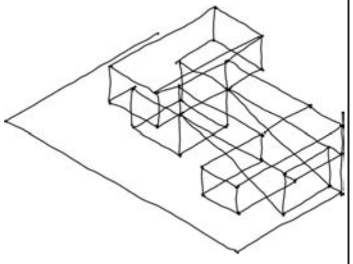
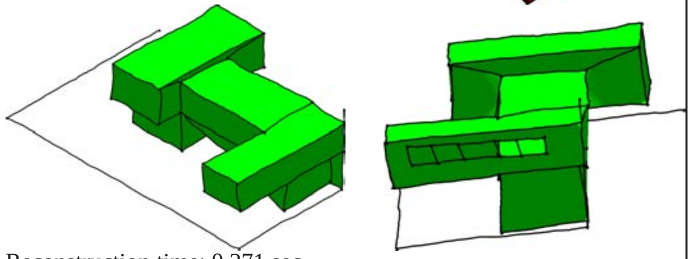
| Original Sketch | Reconstructed Shape |
|---|---|
|  |  Reconstruction time: 0.181 sec |
|  |  Reconstruction time: 0.170 sec |
|  |  Reconstruction time: 0.221 sec |
|  |  Reconstruction time: 0.180 sec |
|  |  Reconstruction time: 0.271 sec |

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

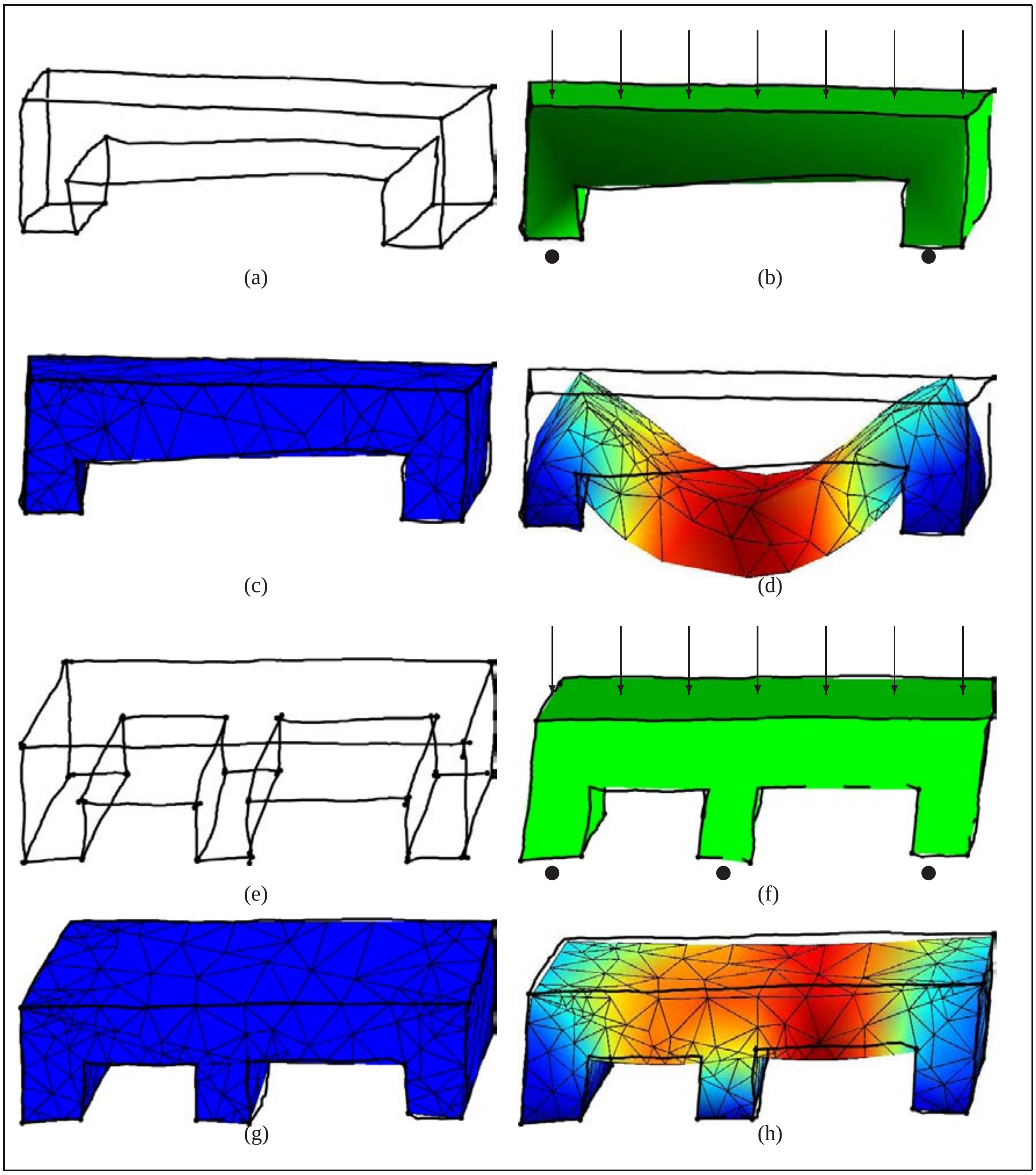


**Figure 5:** *A an example of a simple, iterative reconstruction and analysis session. (a) An initial sketch of a table-like structure with two legs (b) the reconstructed manifold solid. Boundary faces on the bottom of each leg are indicated by dark circles. A 3N downward force is applied to the topmost face. The coefficients of elasticity are relatively large, allowing for a high degree of deformation (c) the object's tetrahedral solid mesh superimposed over the original strokes (d) the tetrahedral mesh following finite element simulation to determine the displacement produced by the 3N force. The original sketch lines are still visible. Dark blue colors indicate nodes with minimal displacement, orange colors indicates nodes with the maximum magnitude displacement (e) a modified version of the original sketch with an additional bracing leg (f) the reconstructed manifold solid with boundary faces on the bottom of each leg. A 3N downward force is applied to the topmost face (g) the object's tetrahedral solid mesh superimposed over the original strokes (h) the tetrahedral mesh following FEA. Note that the additional leg considerably reduces deformation, and that peak deformation occurs on the longer of the two inter-leg sections.*

ing algorithm, which are discussed in detail in [Si05]. Construction of a practical mesh requires limiting the number of samples used to represent each stroke to less than 10 to ensure that the manifold surface is not overly complex, and ensuring that the boundary of each face is consistent with the boundary of any adjacent face so that the manifold surface can be constructed by linking the points along the boundary of each triangulated face. In practice, these restrictions do not limit the usefulness of the FEA simulation as an analysis tool for sketched shapes.

The FEA solution to the elasticity problem requires that at least one face of the manifold solid serve as a boundary that will remain stationary when forces are applied to the solid. Forces may be applied to one or more faces. Faces may be selected directly by clicking with the pen, after which the forces and boundary information are specified with a dialog. The analysis itself is performed using GetFem++ [RP05], a publicly available finite element library. The results of the analysis are superimposed over the strokes making up the sketch, allowing users to very quickly interpret the results and modify the object accordingly.

### 5.1. Results

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 many 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 Figure 4. Examples of non-trihedral vertices can be seen in the first and fourth objects.

An example FEA session is shown in Figure 5. Users were quickly able to design parts and verify their structural integrity under various loads. In practice, the time required to perform the finite element analysis proved to be the limiting factor. Finite element analysis scales to the third order with the number of tetrahedra required to represent the manifold solid, which in turn scales with the complexity of the object. Very simple shapes could be analyzed in several seconds. As complexity increased, however, the iterative analysis became time consuming and the design flow. Though finite element analysis is necessarily computationally complex, analysis time can be decreased by using only coarse approximations to the sketched solid during the iterative design phase.

## 6. Conclusions and Future Work

This paper presented a pen-based sketching system for progressively constructing and analyzing 3D objects using freehand 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 3D object is reconstructed from this sketch, after which the object's faces are identified and triangulated.

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. Users can add additional strokes, or erase strokes, and sketch directly onto the object's faces. The reconstructed object can be submitted to a finite element analysis in order to investigate its physical properties, after which it can be modified if necessary. Other analysis codes can easily be used in place of FEA.

This work was performed to investigate the use of 3D sketching combined with engineering analysis as a conceptual design exploration tool. Future investigation will attempt to further elucidate the types of design tasks for which this form of interface is advantageous. Additional research will also be dedicated to improving the reconstruction algorithm, including extensions to sketches that cannot be described by a single, connected graph, and sketches with multiple axis systems.

## 7. Acknowledgements

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.

## References

[CCCP04] COMPANY P., CONTERO M., CONESA J., PIQUER A.: An optimization-based reconstruction engine for 3D modeling by sketching. *Computers & Graphics 28* (2004), 955–979.

[CPC04] COMPANY P., PIQUER A., CONTERO M.: On the evolution of geometrical reconstruciton as a core technology to sketch-based modeling. In *EUROGRAPHICS Workshop on sketch-based interfaces and modeling* (2004).

[Dav02] DAVIS R.: Position statement and overview: Sketch recognition at MIT. In *AAAI Sketch Understanding Symposium* (Stanford, CA, 2002).

[Fuk98] FUKUI Y.: Input method of boundary solid by sketching. *Computer aided design 20*, 8 (1998), 434–440.

[IMT99] IGARASHI T., MATSUOKA S., TANAKA H.: Teddy: sketching interface for 3d freeform design. In *Proceedings of SIGGRAPH* (August 1999), pp. 409–416.

[KML04] Kang D. J., Masry M., Lipson H.: Reconstruction of a 3d object from main axis system. In *AAAI Fall Symposium Series: Making Pen-Based Interaction Intelligent and Natural* (2004). To Appear.

[LB90] Lamb D., Bandopdhay A.: Interpreting a 3D object from a rough 2D line drawing. In *Proceedings of Visualization '90* (1990), pp. 59–66.

[LS96] Lipson H., Shpitlani M.: Optimization-based reconstruction of a 3D object from a single freehand line drawing. *Journal of Computer Aided Design 28*, 8 (1996), 651–663.

[LS00] Lipson H., Shpitlani M.: Conceptual design and analysis by sketching. *Journal of AI in Design and Manufacturing (AIEDEM) 14* (2000), 391–401.

[LS02] Lipson H., Shpitlani M.: Correlation-based reconstruction of a 3D object from a single freehand sketch. In *AAAI Spring Sumposium on Sketch Understanding* (2002), pp. 99–104.

[Mac73] Mackworth A. K.: Interpreting pictures of polyhedral scenes. *Artificial Intelligence 4* (1973), 121–137.

[PTVF03] Press W. H., Teukolsky S. A., Vetterling W. T., Flannery B. P.: *Numerical Recipes in C++: The Art of Scientific Computing*, second ed. Cambridge University Press, 2003.

[RP05] Renard Y., Pommier J.: Getfem++: a generic finite element library in c++, version 1.7, 2005. http://www-gmm.insa-toulouse.fr/getfem.

[SC04] Shesh A., Chen B.: SMARTPAPER–an interactive and user-friendly sketching system. In *Proceedings of Eurographics 2004* (August 2004).

[SDS98] Stahovich T. F., Davis R., Schrobe J.: Generating multiple new designs from a sketch. *Artificial Intelligence 104* (1998), 211–264.

[She96] Shewchuk J. R.: Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In *Applied Computational Geometry: Towards Geometric Engineering*, Lin M. C., Manocha D., (Eds.), vol. 1148 of *Lecture Notes in Computer Science*. Springer-Verlag, May 1996, pp. 203–222. From the First ACM Workshop on Applied Computational Geometry.

[Si05] Si H.: Tetgen, a quality tetrahedral mesh generator and three-dimensional delaunay triangulator, 2005. http://tetgen.berlios.de/.

[SL97] Shpitlani M., Lipson H.: Classification of sketch strokes and corner detection using conic sections and adaptive clustering. *ASME Journal of Mechanical Design 119*, 2 (1997), 131–135.

[VMS04] Varley P. A. C., Martin R. R., Suzuki H.: Making the most of using depth reasoning to label line drawings of engineering objects. In *9th ACM Symposium on Solid Modeling and Applications SM'04* (2004), Elber G., Patrikalas N.,, Brunet P., (Eds.), pp. 191–202.

[VTMS04] Varley P., Takahashi Y., Mitani J., Suzuki H.: A two-stage approach for interpreting line drawings of curved objects. In *Eurographics Workshop on Sketch-Based Interfaces and Modeling* (2004), Hughes J., Jorge J., (Eds.), pp. 117–126.

[Wei87] Wei X.: *Computer Vision Method for 3D Quantitative Reconstruction from a Single Line Drawing*. PhD thesis, Department of Mathematics, Beijing University, China, 1987. (in Chinese; for a review in English see Wang and Grinstein, 1993).

[WG89] Wang W., Grinstein G.: A polyhedral object's CSG-Rep reconstruction from a single 2D line drawing. In *Proceedings of 1989 SPIE Intelligent Robots and Computer Vision III: Algorithms and Techniques* (1989), vol. 1192, pp. 230–238.

[ZHH96] Zeleznik R., Herndon K., Hughes J.: SKETCH: An interface for sketching 3d scenes. In *Proceedings of SIGGRAPH* (1996), pp. 163–170.