SIACG 2002 - 1st Ibero-American Symposium on Computer Graphics
1-5 July 2002, Guimarães - Portugal

Direct Modeling: from Sketches to 3D Models

Fernando Naya
Dep. Expr. Gráfica Ingeniería, UPV
Camino de Vera, 46020 Valencia
fernasan@degi.upv.es

Joaquim A Jorge
Dep. Engª. Informática, IST
Av. Rovisco Pais, 1049-001 Lisboa
jorgej@acm.org

Julián Conesa
Dep. Expr. Gráfica, UPCT
Dr. Fleming, 30202 Cartagena
julian.conesa@upct.es

Manuel Contero José María Gomis
Dep. Expr. Gráfica Ingeniería, UPV
Camino de Vera, 46020 Valencia
{mcontero,jmgomis}@degi.upv.es

Abstract
In spite of recent advances in Computer Aided Design, Graphical User Interfaces (GUI) are, by and large, still at the stage of the so-called WIMP (Window, Icon, Menu, Pointing device) approach. In recent years, our research team has developed different algorithms in Geometric Reconstruction. The aim of this effort is to obtain an automatic (or, at least, easy-to-use) means to generate 3D models from freehand 2D drawings. This approach serves as the basis to a calligraphic interface, based on freehand sketches and gestures, described as a prototype application capable of modeling special kinds of objects such as normalon and quasi-normalon polyhedra. Using our system users can directly draw the axonometric view of an object to yield a 3D model. While much work remains to be done, the current application already shows gains with respect to more traditional forms of modeling in that it embodies a drawing approach familiar to most draftspeople, who can start modeling relatively complex shapes without much training. Preliminary studies show that our modeling system compares favorably to commercial grade CAD systems both in number of operations required to creating objects and time to accomplish simple modeling tasks.

Keywords
Calligraphic Interfaces, Geometric Reconstruction,

1. INTRODUCTION

Recently, there has been a lot of interest in developing algorithms for interactive geometric construction of 3D models. While most of the activity in this area in the past has been focused in off-line computer vision algorithms, the growing focus on sketches and modeling has brought forth a new emphasis on methods and approaches geared towards interactive applications. To this end, the aim of our research is to develop expeditious ways to construct geometric models. In other words, we want to automatically generate solid and surface models from freehand two-dimensional drawings. As a first approximation, our previous efforts have yielded a reasonably robust geometric modeling program that allows to construct 3D models from axonometric perspective line drawings. This program uses well-formed drawings created by means of a two-dimensional CAD program. In these drawings, the model is defined in pseudo-perspective projection defined by lines which meet precisely at well-defined vertices. These drawings are then exported in a "standard" (DXF) format to the reconstruction application, which performs an optimization task based on perception theory, yielding a boundary representation of the reconstructed object.

Our previous work has yielded a reliable and robust reconstruction core, especially in the case of normalon and quasi-normalon objects (a solid is considered to be a normalon when there are three principal directions, that is, when the line junctions of the figure are oriented in only three directions). The present text describes work at the user interface towards integrating this reconstructor application (REFER) into an interactive working environment, through sketch input using a digitizing tablet and a pen, an approach we have termed *calligraphic interfaces*. These rely on interactive input of drawings as vector information (pen-strokes) and gestures, possibly coupled with other interaction modalities. This environment differs markedly from previous efforts in that speed of execution and feedback are more important than the ability to produce complicated models from vectorized bitmaps in one pass as typical of previous efforts in computer vision. In the next section we briefly describe related work in the field of computer vision and interactive systems. Then we present our system and describe two-dimensional input and three-dimensional reconstruction. Finally, we present examples of models drawn using our system, followed by a preliminary comparison against a commercial CAD system. The last two sections discuss the current work and research directions for future endeavors.

2. RELATED WORK

Despite great advances in CAD systems since the end of the 60s, they still show an excessive stiffness at the first stages of the design process, in which pen-and-paper sketches are the basic tools to express the engineer's creativity [Jenkins93, Ullman90, Goel95].

Although some pioneering work used the light pen as data input devices [Sutherland63], WIMP interfaces (Windows, Icons, Menus, Pointing devices) dominate the commercial CAD packages market today. However, recent research has focused on developing applications that aim at designing person-machine interactive systems as an alternative to the systems available today [Negroponte73][Herot76]. One such approach uses a stylus as input device on a digitizing tablet combined with a LCD display. This aims at providing some of the drawing facilities afforded by conventional pen and paper, commonly used by designers to capture product ideas by sketching. The current generation of powerful computer processors and affordable input/output devices, can justify the feasibility of the new systems. In contrast the early 90s generation of pen computers, plagued by low computing power and expensive devices, at present many portable digital assistants use interfaces based on this type of interaction and a new generation of tablet computers are coming of age that could serve as the tools of choice for designers.

The new generation of applications geared at these new devices, use gestures and pen-input that serve as commands [Rubine92] [Long00]. Pens can also be used to enter continuous mode sketches and freehand strokes. Thus, there is a growing research interest on using freehand drawings and sketches as a way to create and edit 3D geometric models. Within this research area we can distinguish three approaches. One family of systems uses gestures as commands for generating solids from 2D segments. The second approach uses algorithms to generate the geometric reconstruction of an object from sketches that depict the 2D projection of the object. Finally, a third approach combines the two approaches mentioned, to input models through a combination of gesture commands and reconstruction.

Early work has provided many examples of the first approach, which we call *gestural modeling*:

SKETCH [Zeleznik96] basically aimed at architectural forms, in which the geometric model is entered by a sequence of gestures according to a set of conventions, regarding order in which points and lines are entered as well as their spatial relations. For example a primitive of the type Block is defined by three segments starting from the same point. Positive volumes are built in the same direction as the outer normal of an adjacent surface whereas negative volumes are drawn opposite from outer normal. An extension of this approach, the SKETCH-N-MAKE system [Bloomenthal98] aims at machining simple models through numerical control using the Sketch gestural interface for modeling the parts to manufacture.

Quick-Sketch [Eggli97] is a computer tool oriented to mechanical design. It consists of a 2D drawing environment based on constraints. It is also possible to generate 3D models through modeling gestures.

Teddy [Igarashi99], allows free surface modeling using a very simple interface of sketched curves, pockets and extrusions. Users draw the object silhouette using a series of pen strokes. The system automatically proposes a surface using a polygonal mesh whose projection matches the object contour. This system is implemented in Java which makes it can be easily accessible through any Internet browser.

GIDeS [Pereira00] permits data input from a single-view projection or from several dihedral views. When creating object from a single-view perspective, the system uses a simple gesture alphabet to identify a basic set of modeling primitives such as prisms, pyramids, extrusion and revolution shapes, among others. In addition the dynamic recognition of these modeling gestures provides the user with contextual menus and icons to allow modeling user a reduced set of commands.

The second approach, which we call *geometric reconstruction*, uses techniques based on computer vision to build 3D geometric shapes extracted from 2D images, representing their axonometric projections.

The systems we surveyed use two main techniques. The first is based on Huffman-Clowes labeling scheme [Huffman71] [Clowes71]; and the second approach treats reconstruction as an optimization problem [Wang93]. This second approach enables us to obtain what from the point of view of geometry is unrealizable: a 3D model from a single axonometric projection. However, from the psychological point of view it is a well-known fact that humans do not seem to have any problems in identifying 3D models from 2D images. What is more, there seems to exist a general consensus about the "correct" and "simple" models that humans see in each drawing. This is the reason why Geometric Reconstruction, understood as a problem of perception, can be described in terms of mathematical optimization. Perceptual methods are different from other methods in that they try to implement the way in which humans perceive objects using computing sequential language.

This can be done due to the ability to establish similarities between the recursive processes characteristic of optimization and the way the human mind operates. Optimization processes for geometric reconstruction present a fundamental feature which makes them different from other common instances of optimization problems, in that local minima may represent incorrect solutions since they minima may represent incorrect solutions since they may represent 3D models that do not match human visual perception. Some reconstruction browsers have been developed by authors such as Marill, Leclerc, Fischler and Lipson [Marill91, Leclerc92, Lipson96].

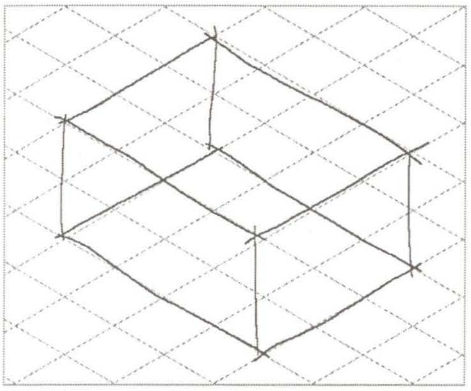

Figure 1: Input Sketch

A third approach for the generation of 3D models from sketches, combines features from the *gestural* and *geometric reconstruction* methods. This is typical of more recently published works:

Digital Clay [Schweikardt98] supports basic polyhedral objects, combined with calligraphic interfaces for data input. The scene is then pre-processed and transferred to a reconstruction engine that uses Huffman-Clowes algorithms to derive three-dimensional geometry. Finally the scene is exported in VRML format.

Stilton [Turner00], although oriented to the field of architecture, presents interesting aspects. First, the calligraphic interface is directly implemented in a VRML environment using image data as texture maps. Second, the reconstruction process is done through optimization based on genetic algorithms.

In comparison to the surveyed work, the application presented here allows interactive reconstruction of *normalon* and *quasi-normalon* type objects from hand input sketches, to yield three-dimensional models. This represents a much richer vocabulary than that of simpler approaches based on extrusion and constructive geometry, which resort to a simple set of basic shapes. In that sense it is arguable that our interface allows greater freedom of modeling without the need to learn special codes for given shapes.

The design paradigm allows composition of complex shapes directly, i.e. from edges drawn as orthogonal projections rather than a composition of simpler shapes built as extrusions, cuts, pockets or holes.

Further, the adoption of a line-based paradigm for interactive reconstruction allows users to directly edit edges and create new vertices into models with rapid feedback.

In the following sections we describe the system operation and philosophy, followed by the methods needed to obtain precise two-dimensional models from sketches which we then convert to three-dimensional models.

3. SYSTEM OPERATION

In order to obtain an application capable to automatically generate 3D models from freehand sketches, we have first developed an application that enables us to generate 3D surface models of the normalon type from the sketch of their axonometric projection. We describe the three-dimensional figure construction algorithm in Section 3.3 [Conesa01]. We note however that this approach requires *precise* and *well-defined* geometry input. Thus we are complementing the three dimensional surface construction with a two-dimensional sketch parsing module, that attempts to convert rough sketches input into rigorous and well connected line drawings. The two subsystems connect in such a way that the 3D model is updated whenever the input sketch is changed, either because new edges are added or deleted. This is true even for invalid input line drawings since the reconstruction system *does not necessarily limit its output to valid solids, only to surface models.* A validation stage will typically be the last step in the creation of a model.

Since most engineering drawings make extensive use of lines that meet at vertices and our application recognizes and processes these entities automatically, both the needs for user training and learning curve are reduced in contrast to current CAD systems. Similarly, edges and segments are removed using a scratching gesture. The application then adjusts sketched lines, to try and maintain a mostly consistent view.

For every change in the sketch, the application executes three steps in sequence. Stroke acquisition, followed by 2D Construction and 3D Reconstruction. Although 2D Construction and 3D Reconstruction steps as defined in the application do not depend on privileged directions in the drawing, we have elected to use an isometric projection for simplicity. However the reconstruction subsystem can work other types of orthogonal cylindrical projections, such as dimetric or trimetric representations.

The only constraint to take into account is that it is necessary to start from a single orthogonal cylindrical projection of the model, which are the single-view representations most commonly used in Engineering.

3.1 Stroke Acquisition

The stage of Stroke Acquisition is responsible for low-level user-application interaction. At this stage data are input via a combination of stylus and digitizing LCD tablet as strokes (sequences of x-y coordinates from pen-down to pen-up) which are then recognized as lines or gestures.

The application has to adequately process the input data, and be capable of extracting the required entities and gestures, neglecting the information which is not necessary. Figure 1 shows the input data generated from the sketch.

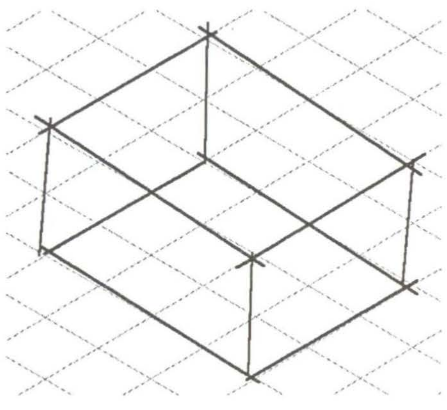
Figure 2: Input entities

From the input strokes recognized, the application only needs the equivalent entities such as lines and vertices extracted from the original sketch, as shown in Figure 2. This is accomplished by translating strokes into vector entities. Rectilinear segments thus constructed are then represented by their start and end points.

To recognize lines and command gestures we have used CALI, a library of software components to develop calligraphic interfaces [Fonseca01]. This library provides a recognizer for elemental geometric forms and gestures in real time, using fuzzy logic and a decision scheme to classify geometric shapes. In this manner we are able to recognize simple geometric shapes, such as triangles, rectangles, circles, ellipses, lines, arrows, etc, and gesture commands, such as delete, move, copy, etc.

Among the different geometric shapes and gestural commands, the application currently selects sketched segments which can be recognized as a geometric form of the type "line" or as a gestural command of the type "delete". In this way, the application analyzes the type of the sketched entity, and if it corresponds to a line or to the command "delete", the entity is processed. If not, the application ignores the new entity as shown in Figure 3.

If the application recognizes a line or the command "delete", these data are fed to the next stage. The application then waits for a new command or line gesture.

3.2 2D Geometric Construction

The aim of this stage is to generate a database from which to generate the 3D model. That is, a pre-processing module is required to transform the data from the table into the necessary format to be used at the stage of 3D Reconstruction. To obtain functional input data, we need to cleanup input data and adjust edges to make sure they meet precisely at common endpoints. This stage includes input sketch filtering techniques which are applied to drawings generated by designers and acquired through the calligraphic interface. These consist mainly of transforming sketches into geometrically consistent figures which can then be used for generating 3D models at the next stage.

But 2D Geometric Construction should not only remove defects caused by designer-made errors. Its main task is

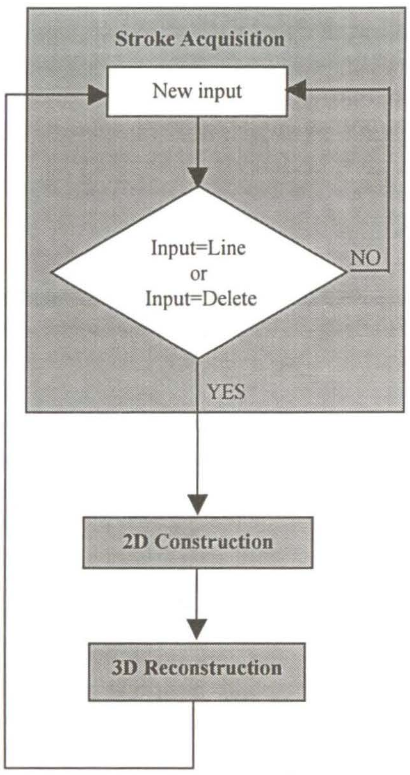

Figure 3: Stages of the application

to filter all defects and errors in initial sketches and which are inherent to their inaccurate and incomplete nature. This problem can be illustrated by drawings in which hand-drawn lines are perceived by humans as parallel although their machine representations are slightly convergent. As another example we have lines which do not meet at precise endpoints, as illustrated by Figure 2. While line endpoints at corners lie "close enough", these endpoints do not define vertices because they are not completely coincident. A fuzzy formulation of these spatial relations, can be used to make sure that e.g. "approximately parallel" maps to "exactly parallel" before applying the three dimensional reconstruction algorithms.

We now look at constructing a two-dimensional representation. At present, the stage of 2D Construction receives as input data from the Stroke Acquisition module, either geometric shapes of type "line", or gesture commands of type "delete". This will be expanded in the near future to include other primitives, such as triangles and quadrilaterals.

When processing a geometric form of type "line", the application has to perform several tasks to create an adequate database for the 3D geometric reconstructor. Among them, we will mention the following:

- Modifying the slope of the new lines that according to the criteria of perception psychology are nearly parallel to one of the three principal axes of the projection (as shown in Figure 4).
- Adjusting the start and end points of each new line in order to make them coincident with *existing* vertices

of the model, while maintaining the appropriate orientation of the edges incident at these vertices.

In order to perform these tasks efficiently, the first step is to classify the new line, depending on its features. We then manipulate its endpoints, so as to match the existing edges of the model and proceed with the stage of 3D Reconstruction.

In order to classify the new line, the line is analyzed following these steps:

The first step consists of checking whether the new line is parallel to any of the principal axes of the sketch, considering a certain tolerance. In the case that the straight line is nearly parallel to one axis, then we adjust one or both endpoints so that the resulting line is now precisely parallel line to one of the three main axis.

The second step looks for vertices close to the line endpoints, again taking into account a proximity tolerance. In the case there are several such vertices, we select the one with the closest to that line endpoint.

For those endpoints of the new line which lie sufficiently near to a vertex, the system records the number of edges to this closest vertex as one of the definition points.

For endpoints of the new line which do not lie close to a model vertex, the system analyzes whether the points are close to an existing edge, accounting for a given tolerance to proximity. If several edges match this criterion, we select the edge which lies closest to the given endpoint.

As we can see the classification process above depends on several tolerance values. These values depend on the length of the new line.

- Tolerance of line parallelism. This tolerance value defines the maximum deviation of the slopes of two straight lines to be considered parallel. This tolerance value is used to define whether a straight line is parallel to any of the principal directions of the drawing.
- Tolerance of vertex proximity. This tolerance value defines the longest distance between the definition points of the new line and the existing vertices, to be considered adjacent. This tolerance value is used to find the Closest Vertex.
- Tolerance of edge proximity. This tolerance value defines the longest distance between the definition points of the new line and the existing edges, to be considered adjacent. This tolerance value is used to find the Closest Edge.

Thus, the new line is classified depending on the values of the following variables:

- Line is Parallel line to one principal axis of drawing (value: TRUE or FALSE).
- Closest Vertex at the start defining point (value: TRUE or FALSE)
- Closest Vertex at the end defining point (value: TRUE or FALSE)
- Number of edges on the Closest Vertex of the start defining point (value: numerical).
- Number of edges on the Closest Vertex of the end defining point (value: numerical).
- Closest Edge at the start defining point (value: TRUE or FALSE)
- Closest Edge at the end defining point (value: TRUE or FALSE)

From this classification, the application can perform the necessary adjustment of the defining points of the new line and of the Closest Vertices and Closest Edges so as to incorporate the line to the input database of the stage of 3D Reconstruction.

If the system receives a gestural command of the type "delete", the application detects the edge(s) that the user wants to delete from the sketch as those intersecting the smallest quadrilateral enclosing the scratching gesture. These are then removed from the drawing, after connectivity information (see below) is updated.

After this stage all the 2D image data are stored in a database. The structure of the stored information is the following:

- List of Vertices. Each of these entities contains the coordinates (x, y) of the vertices that form the model after 2D Construction.
- List of Edges. Each of these entities contains two references to entities of the List of Vertices. Such references define the start and end points of the line. It is also necessary to know which vertex will be the start point, and which vertex will be the end point for the correct development of the reconstruction stage.

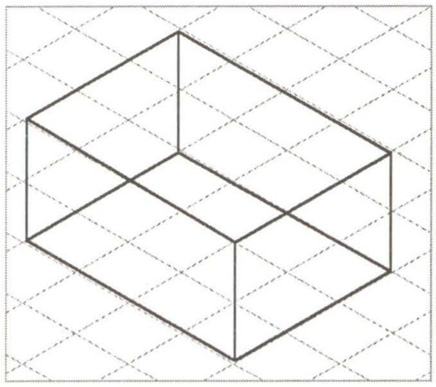

Figure 4: 2D Geometric Construction

Figure 4 shows the entities of the sketch of Figure 1 after 2D Construction. This vectorized image is suitable for the following stage.

3.3 3D Geometric Reconstruction

At this stage, a 3D geometric model is obtained from vector data, using the geometry information from the object "*implicitly contained*" or perceptually projected in the initial figure.

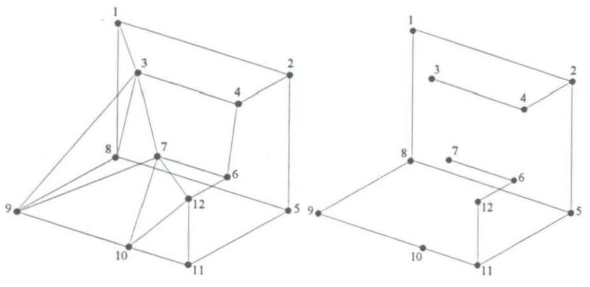

Figure 5: Normalon equivalent to a quasi-normalon

As a starting point, the X and Y axes of the model coordinate system are taken from the model projection plane. That is, each vertex of the model is assigned the same coordinates (x, y) as those of its corresponding projection. The z coordinates of the image are initially set to zero, and the z coordinates of each model vertex are defined as construction variables. As a consequence, each vertex of the model must lie on a projecting line, which is perpendicular to the image plane, since the projection is assumed to be orthogonal.

In short, our solution space is defined by the values assigned to the z coordinates of each of the image vertices.

In our case, we work with normalon-type polyhedra. To recap, we consider as normalon polyhedra those figures whose edges are parallel to one of three principal directions. Normalon polyhedra can be automatically and directly reconstructed without having to resort to optimization, which permits a fast reconstruction process as well as to work on-line. The reconstructing process consists of swelling the two-dimensional image.

It is also possible to apply this procedure to polyhedra other than normalons, provided that we can evaluate the position of all the vertices, using only edges parallel to any of the three principal axes. In other words, such models need to fulfill the condition that removing all edges which are non-parallel to three principal directions can be done without the loss of vertices (Figure 5). We use the term *quasi-normalon* to designate models which satisfy this condition.

Let's note that in quasi-normalon images the principal directions are those directions which verify a set of constraints:

1. Projecting constraint: the difference between the maximum and minimum angle of the edges considered as principal directions should be greater than 90°, to correspond with an axonometric projection.
2. Topologic constraint: There should exist at least one vertex on which three edges parallel to directions considered as principal directions coincide.

The application of the swelling method to axonometric projections permits the direct reconstruction of normalon and quasi-normalon models, with no need for time-consuming optimization processes and obviating the need for *extra* interaction between the user and the system [Conesa99]. After this stage the application generates a surface model of the part which we previously sketched.

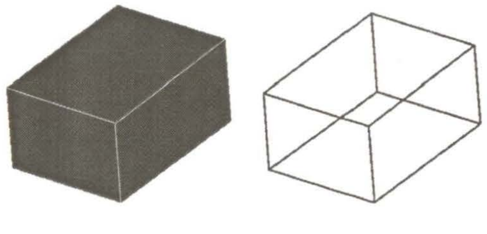

Figure 6: 3D Geometric Reconstruction

Once we have acquired the three-dimensional model, it is possible to generate its representation. The application allows us to view the resulting geometrical part either as a wire-frame or a surface model. Figure 6 shows these two different displays for the model sketched in Figure 1. In addition, it is also possible to manipulate the view of the reconstructed model.

Whenever the user enters changes to the input sketch, the application executes the stage of 3D Reconstruction, which allows the user to see immediately how the design evolves, even through intermediate stages. Thus, while modifying the object sketch, the user can see the 3D result produced. Sometimes it is desirable to turn off this behavior if the intermediate models prove confusing.

4. EXAMPLES

In the current section we illustrate some examples of models built with our prototype application.

Figure 7 presents different objects modeled with our system. On the left side we show the two dimensional images corresponding to the user's sketches. On the right side we present the equivalent surface model as generated from the sketch. While the model depicted in Figure 7 b) shows a normalon typology, all the other models of this diagram are quasi-normalon.

As is well known, the axonometric projection of a model may correspond to two different three-dimensional models; that is, from a single view we can obtain either a model or its Necker converse, as shown in Figure 8. By default, the application displays models b) and e). This assumption follows the psychological perception of small objects with respect to human size, when watched from top to bottom, although this behavior can be changed to suit user's preferences.

5. INTERACTION

Although some of the results presented here are preliminary, we have used them to assess the viability of new approaches to develop graphical interfaces for modeling three dimensional geometric representations of objects.

We have checked the number of elementary operations required to create some of the (admittedly simple) models created by our application as compared to those required by a commercial CAD application, operated by an expert user with three years experience using the package. The results are presented in Table 1. As we can see, the concept of elementary operations differs markedly between the two systems.

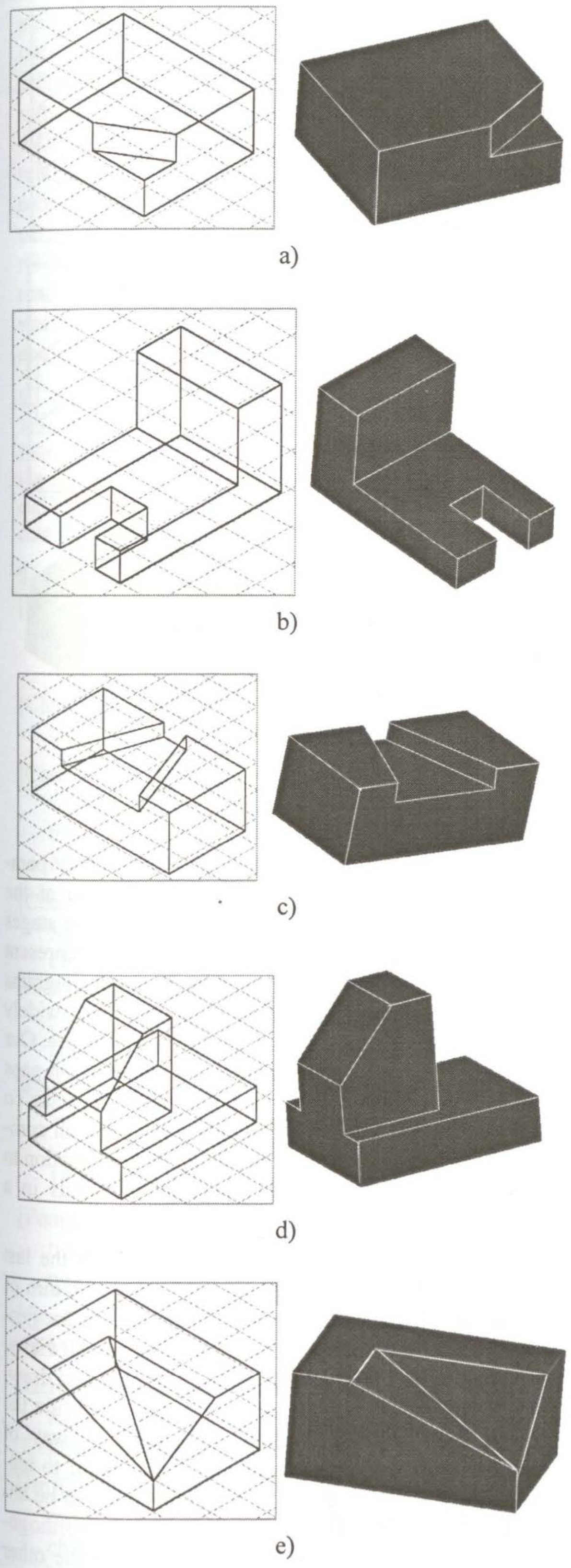

Figure 7: Normalons and quasi-normalons

Conventional CAD systems are organized around menus, command selections, view manipulation, geometry input through discrete sequences of data points occasionally complemented with direct manipulation and attribute (scale, feature, etc.) editing.

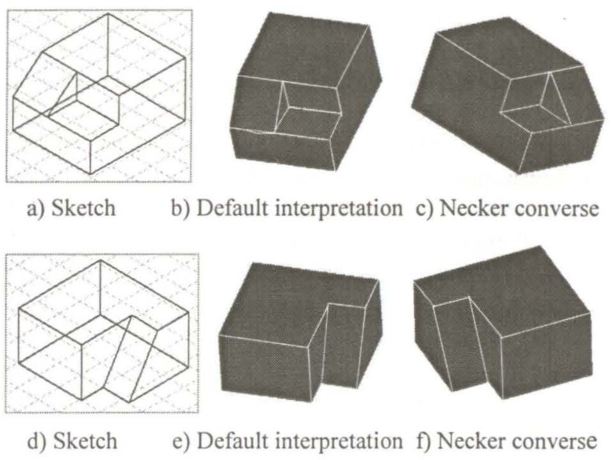

Figure 8: Necker conversion

		Fig.7 c)	Fig.7 d)	Fig.9
Our System	# strokes	22	26	19
	# Erase cmds.	2	1	1
	# Camera Ops	3	3	3
	Total	**27**	**30**	**23**
Commercial CAD	# Menu Sels.	31	33	33
	# Data Points	7	8	6
	# Camera Ops	3	3	3
	Total	**41**	**44**	**42**

Table 1: Operations required with our approach and a commercial system

Our system currently provides three types of elementary constructs: stroke input (a continuous sequence of points entered in a single interaction from pen-down to pen-up), erase (scratch) gesture and view manipulation. Lines implicitly connect at vertices with vertices being created as needed. We have also conducted informal usability tests with a number of non-expert users to assess learnability and simplicity of use.

Preliminary data show that the drawing approach is easy to explain and to learn, with users being able to apply learned drawing skills to create simple models after less than five minutes training. This contrasts favorably with the amount of coaching required to accomplish the same with the commercial system. User acceptance was high, reflecting a better match to the task. This is probably because CAD systems require that users spend most of the time navigating menus, which accounts for roughly 75% of the elementary operations, *for experienced users*, as can be readily seen from Table 1. Only a small fraction of all commands are devoted to actual geometry input (16% if we discount view manipulation). While we cannot argue that the learning curve of calligraphic modeling systems such as ours is less steep than that of conventional approaches, the data collected thus far show that it is possible to make the functionality accessible in a more familiar way, due to the ready analogies with the pencil-and-paper model. Further, even from the results gathered with our simple prototype, we can argue that the expressiveness in calligraphic interfaces tends to be higher than that of WIMP approaches in that a gesture can indicate which objects are affected, what command to perform and where and how to show the results in one single interaction. Further, the results in Table 1 were achieved with a very simple command set. It is arguable that even more impressive gains will be possible with a richer command set, using polylines and other two dimensional single-stroke figures.

Figure 9 shows how users can modify a simple model to arrive at more complex shapes in a controlled manner, while using an edge-modification paradigm. It illustrates how users can change shapes by adding, removing and changing edges from the base sketch, which are reflected in the resulting three-dimensional geometry without any explicit or implied solid model operations, extrusions or pockets, allowing users to focus on the drawing task instead on constructing or manipulating the geometric representation details.

6. FUTURE WORK

Our final aim is to develop a computer application that expeditiously allows users to generate 3D models in standard formats exportable to commercial CAD packages from 2D freehand sketches. We use a *hybrid approach* to generate models that combines the gestural commands with geometric reconstruction algorithms to generate normalon or quasi-normalon objects which are used as new modeling primitives. This corresponds to a conceptual reduction in command set size as compared to pure gesture systems, where complex shapes are built from a vocabulary of basic extrusion models.

In order to reach our final goals, it would be interesting to integrate geometric reconstruction as presented here within a sketching application such as the GIDeS environment to allow modeling more complex primitives to complement GIDeS gesture alphabet.

Among future developments, it could be interesting to export reconstructed models in other formats. We have developed a module for writing VRML 2.0 and are coding a new module to allow exporting models in STEP format (ISO 10303), according to application protocols 203 and 214.

Another important issue would be to extend the typology of polyhedra reconstructed by the application, adding other forms such as prismatic polyhedra (polyhedra bounded by a prismatic surface and two planes, generally parallel, that cut all the edges of the prismatic surface generating parallelograms), or pyramidal polyhedra, in which all faces except one meet at one point.

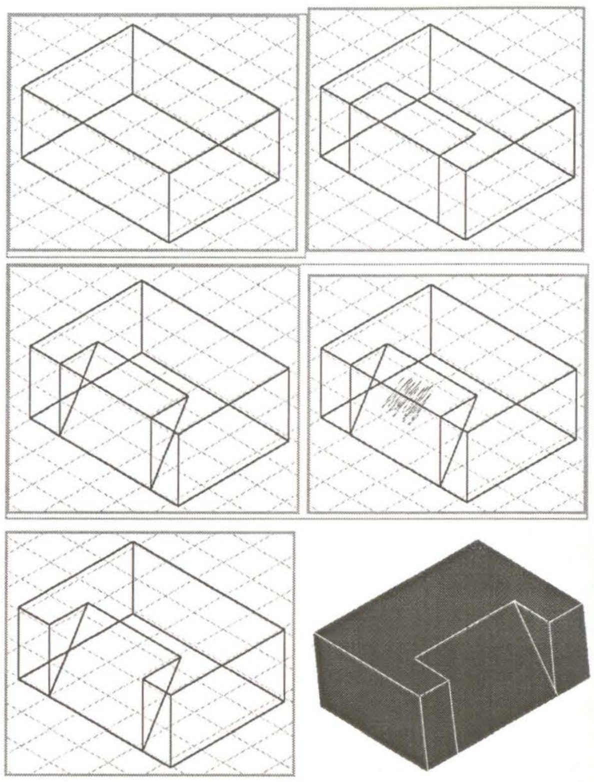
Figure 9: Editing a simple model

7. CONCLUSIONS

Current CAD systems are still based on the WIMP paradigm, which makes them ill-suited for adoption at the early stages of product and model design. At these stages pencil-and-paper sketches are better suited to represent the creative ideas in a fast way. Sketches and diagrams are actually the natural communication techniques widely used by engineers, composers, architects, artists, etc. Our goal is to try and bridge the mental and articulation gaps that make current CAD systems unsuited to the task. To this end we have presented a calligraphic approach combining simple commands with geometric reconstruction to illustrate how to accomplish some of these goals in a straightforward manner.

From the experience acquired by some of us in the last years in the field of geometric reconstruction, the aim of this effort is to develop an automatic application for generating geometric models from two-dimensional views. The present work focuses on freehand sketches and drawings as a way to obtain 3D geometric models. While the prototype application provides an expeditious way of developing such interfaces, several issues related to ambiguous drawings in isometric perspective need to be addressed. These will be tackled by allowing other orthogonal perspectives, e.g. dimetric to be used. On the other hand a better exploitation of ambiguity as highlighted by the GIDeS [Pereira00] system can be put to good use here.

While much work remains to be done, preliminary results garnered from experimenting with the current prototype are very encouraging. The calligraphic approach to modeling which allows users to focus on the drawing task rather than on the subtleties of geometric representations seems to offer great advantages both on lesser number of steps as well as a more familiar approach, as compared with current modeling systems, since it builds on skills related to drawing sketches on paper. Further, this approach also bears the promise of a smoother learning curve as compared to conventional CAD systems. The results obtained so far promising as they are, constitute a ready incentive to extend and improve our approach towards more sophisticated, yet more natural modes of modeling with computers.

8. ACKNOWLEDGEMENTS

This work was performed partially at INESC-ID with support from FCT and EC Grant IST-2000-28169 and "Stays of Teaching and Research Staff of the Polytechnic University of Valencia in Prestigious Research Centers" within the program of Incentives to Research of the U.P.V.

9. REFERENCES

[Bloomenthal98] Bloomenthal K., Zeleznik R.C. et al. SKETCH-N-MAKE: Automated Machining of CAD Sketches. Procs. of ASME DETC'98, 1-11, 1998.

[Clowes71] Clowes M.B. On Seeing Things. Artificial Intelligence, 2, 79-116, 1971.

[Conesa01] Conesa J. Reconstrucción Geométrica de sólidos utilizando técnicas de optimización. PhD thesis, Polytechnic University of Cartagena (Spain), November 2001.

[Conesa99] Conesa J., Company P., Gomis J.M. Initial Modeling Strategies for Geometrical Reconstruction Optimization-Based Approaches. Proceedings,11th ADM International Conference on Design Tools and Methods in Industrial Engineering. Palermo (Italia), December 1999.

[Eggli97] Eggli L., Hsu C., Brüderlin B.D., Elber G. Inferring 3D Models from Freehand Sketches and Constraints. Computer-Aided Design, 29(2), 101-112, 1997.

[Fonseca01] Fonseca M., Jorge J. Experimental Evaluation of an On-Line Scribble Recognizer. Pattern Recognition Letters, 22 (12), 1311-1319, 2001

[Goel95] Goel V. Sketches of Thought. Cambridge, MA: MIT Press, 1995.

[Herot76] Herot C. Graphical Input through Machine Recognition of Sketches. ACM SIGGRAPH Computer Graphics, 10 (2), 97-102, 1976.

[Huffman71] Huffman D.A. Impossible objects as nonsense sentences. In Meltzer B., Michie D. eds. Machine Intelligence No 6, Edimburgo UK. Ediburgh University Press, 295-323, 1971.

[Igarashi99] Igarashi T., Matsuoka S., Tanaka H. Teddy: A Sketching Interface for 3D Freeform Design. ACM SIGGRAPH Conference Proceedings, 409-416, 1999.

[Jenkins93] Jenkins D.L., Martin R.R. The importance of free-hand sketching in conceptual design: automatic sketch input. Design Theory & Methodology (DTM 93), Hight T.K. y Stauffer L.A.Eds., 115-128, ASME Vol. DE-53, 1993.

[Leclerc92] Leclerc Y., Fischler M. An Optimization-Based Approach to the Interpretation of Single Line Drawing as 3D Wire Frames. International Journal of Computer Vision, 9 (2), 113-136, 1992.

[Lipson96] Lipson H., Shpitalni M. Optimization-Based Reconstruction of a 3D Object from a Single Freehand Line Drawing. Computer Aided Design, 28 (8), 651-663, 1996.

[Long00] Long A.C., Landay J.A., Rowe L.A., Michiels J. Visual Similarity of Pen Gestures. CHI'00 Proceedings, 360-367, 2000.

[Marill91] Marill, T. Emulating the Human Interpretation of Line-Drawings as Three-Dimensional Objects. International Journal of Computer Vision, 6 (2), 147-161, 1991.

[Negroponte73] Negroponte N. Recent advances in sketch recognition. Proceedings of the AFIPS 1973 National Computer Conference, 663-675, 1973.

[Pereira00] Pereira J., Jorge J., Branco V., Nunes F. Towards calligraphic interfaces: sketching 3D scenes with gestures and context icons. WSCG'2000. Conference Proceedings, Skala V. Ed., 2000.

[Rubine92] Rubine D. Combining gestures and direct manipulation. Procs. ACM CHI'92 Conf. on Human Factors in Computing Systems, 659-660, 1992.

[Schweikardt98] Schweikardt E., Gross M.D. Digital Clay: deriving digital models from freehand sketches. ACADIA '98, Seebohm T. and Wyk S. V. eds., Quebec City, Canada, 202-211, 1998.

[Sutherland63] Sutherland I.E. Sketchpad: a man-machine graphical communication system. Proc. Spring Join Computer Conference. AFIPS. 329-346, 1963.

[Turner00] Turner A., Chapman D., Penn A. Sketching space. Computers and Graphics, 24 (12), 869-879, 2000.

[Ullman90] Ullman D.G., Wood S., Craig D. The importance of drawing in the mechanical design process. Computers and Graphics, 14 (2), 263-274, 1990.

[Wang93] Wang W., Grinstein G. A Survey of 3D Solid Reconstruction from 2D Projection Line Drawing. Computer Graphics Forum, 12 (2), 137-158,1993.

[Zeleznik96] Zeleznik R.C., Herndon K.P., Hughes J.F. SKETCH: An interface for sketching 3D scenes. SIGGRAPH'96, 163-170, 1996.