# Parametric Freehand Sketches

Ferran Naya[1], Manuel Contero[1], Nuria Aleixos[2], Joaquim Jorge[3]

[1] DEGI - ETSII, Universidad Politécnica de Valencia, Camino de Vera s/n,
46022 Valencia, Spain
{fernasan, mcontero}@degi.upv.es
[2] Departamento de Tecnología, Universidad Jaume I, Avda. Sos Baynat
12071 Castellón, Spain
naleixos@tec.uji.es
[3] Computer Science Department, IST/UTL, Av. Rovisco Pais
1049-001 Lisboa, Portugal
jorgej@acm.org

**Abstract.** In this paper we present the 2D parametric freehand sketch component of an experimental prototype called GEGROSS (GEsture & Geometric ReconstructiOn based Sketch System). The module implements a gesture alphabet and a calligraphic interface to manage geometric constraints found in 2D sections, that are later used to perform modeling operations. We use different elements to implement this module. The geometric kernel stores model data. The constraint manager 2D DCM handles restrictions. Finally, we use the CALI library to define gestural interfaces. In this paper we present a strategy for integrating these tools, and a calligraphic interface we developed to provide dimensional controls over freehand sketches. Our system allows users to build simple sketches composed by line segments and arcs, which are automatically tidied and beautified. Proportional and dimensional information over sketched parts is provided by handwriting their corresponding sizes.

## 1 Introduction

The WIMP (Windows, Icons, Menus and Pointing) interface paradigm dominates user interfaces in most CAD applications. Recently, work on sketch-based modeling aims at a paradigm shift to change the way geometric modeling applications are built, in order to develop user-centric systems rather than systems that are organized around the details of geometry representation. New hardware devices such as Tablet-PCs open new opportunities to experiment with different user interfaces, based on drawing and sketching using a digitizing tablet and a pen, an approach we have termed *calligraphic interfaces*. These rely on interactive input of vector information (pen-strokes) and gestures instead of menu selection and point input as their main organizing principle.

Our paper describes work on the 2D parametric freehand sketch module of GEGROSS. This module implements a gesture alphabet and a calligraphic interface to manage the geometric constraints found in 2D sections, that later are used to perform geometric modeling operations.

Our sketching system aims at providing an intelligent modeling tool to supports visual thinking in the conceptual design stage of product development, where designers usually employ freehand sketches to record different design alternatives. Integrating both gesture and reconstruction modeling capabilities, we offer a very simple interface to create geometric models, with an interesting possibility: “dimensional control”. For users who want to explore design alternatives, or adjust a sketch to satisfy a set of dimensional constraints, we provide parametric capabilities to edit 2D freehand sections. In this way, users can dynamically drag a section edge to modify its geometry, or they can impose dimensional constraints by drawing the corresponding dimension label and writing its value. To this end, handwritten text recognition capabilities provide a very simple method to change dimension values. Moreover, geometric constraint manipulation is possible via the gesture alphabet we developed.

Our approach differs from previous sketch systems in the integration of both gestural and reconstruction methods. These define a new kind of systems we have called “hybrid” modelers that we think provide greater flexibility and modeling capabilities over other techniques.

## 2 Related Work

Form surveying the literature we identify two main approaches to sketch-based modeling. The first is based on the calligraphic interfaces that use gestures and pen-input as commands [1, 2, 3]. These *gestural modeling* systems rely on gestures as commands for generating solids from 2D sections. One good example is Sketch [4], where the geometric model is entered by a sequence of gestures according to a set of conventions regarding the order in which points and lines are entered as well as their spatial relations. Quick-Sketch [5] is a system oriented to mechanical designs that consists of a 2D drawing environment based on constraints. From these it is possible to generate 3D models through modeling gestures. Another system, Teddy [6] is oriented to free-form surface modeling using a very simple interface of sketched curves, pockets and extrusions. Users draw silhouettes through a series of pen strokes and the system automatically proposes a surface using a polygonal mesh whose projection matches the object contour. In [20] the authors present an improved sketch-based mesh extrusion method. GIDeS [7] allows data input from a single-view projection. In addition the dynamic recognition of modeling gestures provides users with contextual menus and icons to allow modeling using a reduced command set. The ISID [21] system can accept strokes, just like drawing on paper with a pen freely. ISID transforms strokes into edit commands or beautifies them into a complete design schematic. Based on the intelligent recognition technique and integrated modeling templates, ISID tries to simplify and optimize sketching to provide a natural, intensive and tangible interface for conceptual design.

The second approach, which we call *geometric reconstruction*, uses computer vision algorithms to reconstruct geometric objects from sketches that best match a two dimensional projection. The systems we surveyed use two main techniques. The first is based on Huffman-Clowes labeling scheme [8], [9]. The second approach handles reconstruction as an optimization problem [11]. This enables us to obtain what is unrealizable, from the point of view of geometry: a three-dimensional model from a single projection. However, it is well known by psychologists that humans can identify 3D models from 2D images by using a simple set of perceptual heuristics [12]. Authors such as Marill [13], Leclerc and Fischler [14], and Lipson and Shpitalni [15] provide interesting references in the development of this field. The Digital Clay system [16], which supports basic polyhedral objects in combination with a calligraphic interface for data input, uses Huffman-Clowes algorithms to derive three-dimensional geometry. Stilton [17] uses a calligraphic interface directly implemented on a VRML environment. Its reconstruction process uses an optimization technique based on genetic algorithms. Finally, CIGRO [19] which supports drawing constrained polyhedral objects (normalon and quasi-normalon ones), and, in combination with a calligraphic interface for data input, uses an axonometric inflation engine to derive three-dimensional geometry.

## 3 Overview of Operations

In contrast to surveyed work, the GEGROSS application integrates both gestural and reconstruction based approaches. This extends the capabilities of our previous reconstruction modeling system CIGRO [19], in order to support gestural modeling. At its current level of development, system behavior is explicitly controlled by users, who can alternate between gestural or reconstruction operation modes. A novel contribution of this technique is to provide support for defining freehand parametric sketches. Gestural modeling is organized in two stages. First, we must define a 2D profile. Then, using a combination of modeling gestures, we build new geometry. In this paper we will describe the 2D parametric freehand sketch module. This could be extended to implement a stand-alone 2D parametric drawing application.

The main objective that guided the interface design for this module was to provide a familiar drawing strategy that takes into account the way engineers create sketches and technical drawings. For the first version of our system, we imposed some sketch creation rules, to simplify implementation tasks. This sped up the development of the first application prototype, allowing us to test the application with engineering students and engineering companies' technical staff to get their feedback.

The first condition we imposed to define sketching was to use stylus pressure level as the main criterion to distinguish between auxiliary information, such as center lines, construction lines, dimensions, editing gestures, etc. and geometric primitives, such as line segments, arcs and circles. Drawings made applying high pressure on the stylus signify intended for geometry input, while soft pressure conveys auxiliary information.

The second condition is related to sketch order. Some multi-stroke entities must be drawn sequentially in time. For example, dimension elements must be sketched in a "logic" order: first we draw extension lines, then dimension lines and arrows, and finally dimension text is drawn last.

We designed the user interface to minimize interaction with menus or icons in an attempt to emulate the traditional use of pen and paper by resorting to drawing conventions used in technical drawings.

### 3.1 Gesture recognition

The current implementation of GEGROSS processes strokes drawn directly on the screen of a Tablet-PC or LCD tablet, as captured by Microsoft Windows XP Tablet-PC Edition SDK. This API allows retrieving additional information such as the pressure applied on the stylus tip at each point of the stroke. Raw strokes are then processed by an enhanced analyzer based on the CALI library [3]. This library recognizes elemental geometric forms and gestural commands in real time, using fuzzy logic. Recognized gestures are inserted into a list, ordered according to a computed degree of certainty. This list is then returned to the application. CALI recognizes the elementary two-dimensional geometric shapes, such as triangles, rectangles, circles, ellipses, lines, arrows, etc. and some gestural commands, such as delete, move, copy, among others.

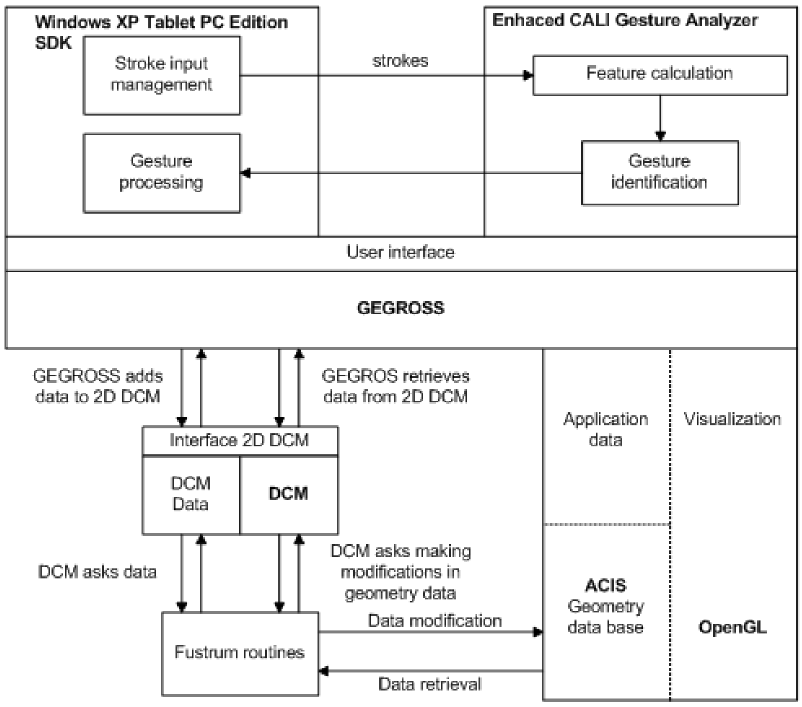


**Fig. 1.** Structure of the freehand parametric sketch module in GEGROSS.

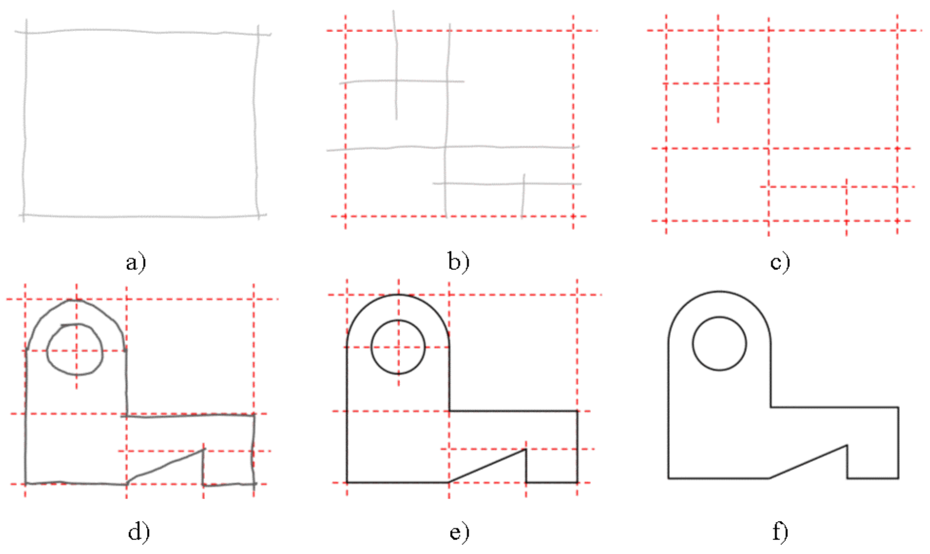


**Fig. 2.** Sketching sequence in GEGROSS. Grey color represents raw strokes corresponding to auxiliary construction lines. Cyan dashed lines represent adjusted auxiliary lines. Black raw strokes in d correspond to real geometry that is snapped to auxiliary-line skeleton depicted in c.

Currently, GEGROSS only supports strokes which can be recognized as entities of *line, arc or circle* types or as gestural commands of the *delete* or *geometric constraint* classes. Through a user-defined pressure threshold, strokes are classified either as real (geometric) or auxiliary (construction) entities.

### 3.2 Sketching procedure

In drafting, most engineers draw a set of auxiliary lines (see Figure 2.a to 2.c) to define main geometric features of a section, using this framework as a drawing template. This sketch is refined by applying pressure with the pencil and drawing over the previous template (see example in Figure 2.d). GEGROSS implements this sketching paradigm by allowing oversketching of real geometry over auxiliary lines, which provide a set of references for snapping purposes.

The user interface supports three visualization modes: rough sketch, beautified sketch and parametric sketch. The user can alternate between views via visualization buttons. Depending on intent, users can choose the adequate visualization mode. If they want only to explore design alternatives, they can do so in rough sketch mode. This is equivalent to drawing a sketch on paper. To work with automatic beautified sketches they can press the corresponding push button. When they want to perform dimensional analysis, they can switch to the parametric sketch mode. In this later mode it is possible to visualize geometric constraints and dimensions found by applying the auto-constraint and auto-dimensioning capabilities of the parametric engine.

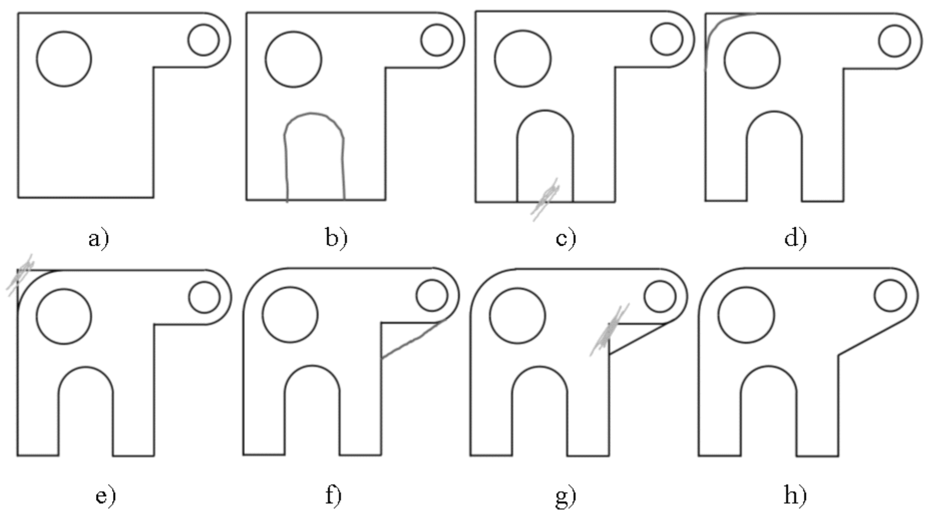


**Fig. 3.** Refining sketch geometry from simple shapes

Drawing entities can be removed using a scratching gesture. This not only allows errors to be corrected but it also enables more complicated shapes to be drawn by refining "simpler" forms as illustrated in Figure 3. When users draw a scratching gesture, the application identifies the entities that they want to delete as those intersected by the smallest quadrilateral enclosing the gesture.

## 3.3 Line Drawing Beautification

If users prefer to work with interactively beautified sketches, GEGROSS provides the corresponding visualization mode (see Figure 3). Online beautification provides immediate feedback, since it operates online as a user draws the sketch. This concept of beautification bears some relation to the drawing beautification proposed by Igarashi [6], [18]. Also it exploits the auto-constraining capabilities of the parametric engine in a way transparent to the user. Both parametric and beautified models are continuously updated by the system, regardless of the current visualization mode.

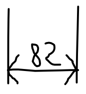
With the help of the 2D DCM parametric engine, from D-cubed [9], we clean up input data in several ways. These include adjusting edges to make sure they meet precisely at common endpoints to get geometrically consistent figures. We also filter all defects and errors of the initial sketches that are inherent to their inaccurate and incomplete nature. In beautification visualization mode, users have immediate feedback (see Figures 2 and 3). To this end we have implemented a number of drawing aids.

**Automatic line slope adjustment** checks whether newly-created lines are either vertical or horizontal or parallel to other line by considering a slope tolerance.

**Vertex point snap** looks for vertices close to geometric entities' endpoints, within a vertex proximity tolerance. Should there be several such vertices, we select the one closest to the candidate entity endpoint.

**Vertex on entity snap and automatic line breaking** applies to endpoints of the new entity which do not lie close to a model vertex. Our system analyzes whether the points are close to an existing edge, taking into account a given edge proximity tolerance. If several edges match this criterion, we select the edge that lies closest to the given endpoint. Then, the selected edge is broken at the contact point; in order to simplify later delete operations, as depicted in figure 3.g.

Snapping and adjustments are performed in real time. Users can see the effect immediately if they are working in beautification or parametric mode. A general tolerance parameter controls beautification actions. This is because some users prefer a less automatic drawing control.

| | |
|---|---|
| Parallel | Perpendicular |
| // | ⊾ |
| Vertical | Horizontal |
| V | H |
| Tangent | Concentric |
| t | ◎ |
| Dimension | Symmetric |
|  | ¦ |

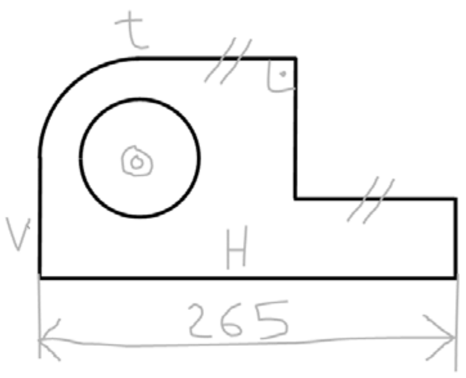


**Fig. 4.** Constraint alphabet and its use in describing a section.

## 3.3 Parametric Edition

This optional stage for section definition is intended to provide geometric control over the sketch. For this purpose we adopted the 2D DCM parametric engine. Using the auto-constraint and auto-dimension features provided by the parametric engine we make sure that the section is adequately constrained and dimensioned at any stage of the section creation. Two visualization buttons enable users to visualize constraints and dimensions. Here we distinguish between "automatic" and "user defined" constraints and dimensions. The former are provided by the system and represented in different color. The latter are sketched by user. Geometric constraints are represented according to the symbols presented in Fig. 4. These symbols can be written by the user to impose some desired constraint. Also, they can be deleted using the scratch gesture. User defined dimensions are sketched using a logic order to simplify parsing.

Extension lines, dimension line, arrows and dimension value are drawn in sequence using low pencil pressure. A dimension sketched without a value, is considered a measure, which the system provides automatically. If a user wants to change this value, he/she deletes the text and then writes down the new value (see Figure 5 for an example sequence). Handwritten recognition is provided by Windows XP Tablet PC Edition system. In this way, we provide a natural way of specifying the desired dimensions on a sketch. Also, users can drag any geometric entity in a section. The system presents the sketch evolution in real time while preserving geometric constraints.

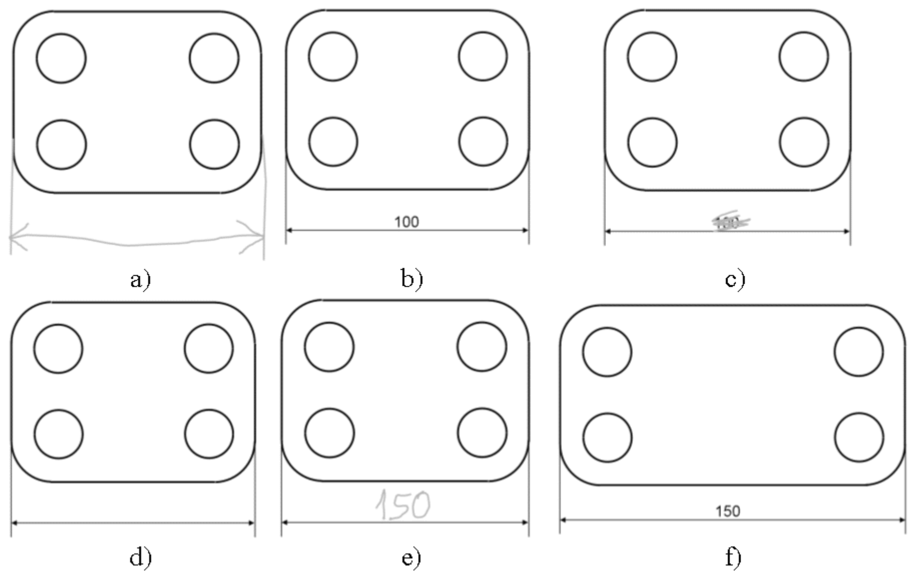


**Fig. 5.** Example of parametric edition in GEGROSS

## 4 Conclusions and Future Work

We have described an approach to input two dimensional sections through the gesture modeling capabilities of our GEGROSS application. The main objective of this work has been to provide dimensional control over the sections in a easy and natural way. With this purpose, handwritten dimensions offer a natural method, that is known by any engineer. Preliminary usability tests have shown encouraging results. Users with an engineering background find the system behavior very natural. Also, the learning process to manage the application is both simple and fast, since it matches conventional drafting techniques. In the near future we aim at implementing ISO 10303 capabilities to support parametric sections, as those defined in part 108 “Parameterization and constraints”.

## Acknowledgments

This work was supported in part by European Commission grant IST-2000-28169 (SmartSketches) and grant CTIDIB/2002/51 from Spanish Generalidad Valenciana.

## References

1. Rubine, D.: Combining gestures and direct manipulation. Proceedings ACM CHI'92 Conference Human Factors in Computing Systems (1992) 659-660
2. Long, A.C., Landay, J.A., Rowe, L.A., Michiels, J.: Visual Similarity of Pen Gestures. Proceedings of Human Factors in Computer Systems (SIGCHI), (2000) 360-367
3. Fonseca, M., Jorge, J.: Experimental Evaluation of an On-Line Scribble Recognizer. Pattern Recognition Letters, **22** (12), (2001) 1311-1319
4. Zeleznik, R.C., Herndon, K.P., Hughes, J.F.: SKETCH: An interface for sketching 3D scenes. SIGGRAPH’96 Conference Proceedings (1996) 163-170
5. Eggli, L., Hsu, C., et al.: Inferring 3D Models from Freehand Sketches and Constraints. Computer-Aided Design, **29** (2), (1997) 101-112
6. Igarashi, T., Matsuoka, S., Tanaka, H.: Teddy: A Sketching Interface for 3D Freeform Design. ACM SIGGRAPH99 Conference Proc. (1999) 409-416
7. Pereira, J., Jorge, J., Branco, V., Nunes, F.: Towards calligraphic interfaces: sketching 3D scenes with gestures and context icons. WSCG’2000 Conference Proc. Skala V. Ed. (2000)
8. Huffman, D.A.: Impossible Objects as Nonsense Sentences. In: Meltzer B., Michie D. (eds.) Machine Intelligence, No. 6, Edimburgh UK. Edinburgh University Press (1971) 295-323
9. D-Cubed, Cambridge, UK, http://www.d-cubed.co.uk/
10. Clowes, M.B.: On Seeing Things. Artificial Intelligence, **2**, (1971) 79-116
11. Wang, W., Grinstein, G.: A Survey of 3D Solid Reconstruction from 2D Projection Line Drawing. Computer Graphics Forum, **12**(2), (1993) 137-158
12. Hoffman, D.: How we create what we see. Visual Intelligence, Norton Pub., **2**, (2000)
13. Marill, T.: Emulating the Human Interpretation of Line-Drawings as Three-Dimensional Objects. International Journal of Computer Vision, **6**(2), (1991) 147-161
14. Leclerc, Y., Fischler, M.: An Optimization-Based Approach to the Interpretation of Single Line Drawing as 3D Wire Frames. Int. Journal of Computer Vision, **9**(2), (1992) 113-136
15. Lipson, H., Shpitalni, M.: Optimization-Based Reconstruction of a 3D Object from a Single Freehand Line Drawing. Computer Aided Design, **28**(8), (1996) 651-663
16. Schweikardt, E., Gross, M.D.: Digital Clay: deriving digital models from freehand sketches. Automation in Construction, **9**, (2000) 107-115
17. Turner, A., Chapmann, D., and Penn, A.: Sketching space. Computers & Graphics, **24**, (2000) 869-879
18. Igarashi, T., Matsuoka, S., Kawachiya, S., Tanaka, H.: Interactive Beautification: A Technique for Rapid Geometric Design, UIST'97, (1997) 105-114
19. Contero, M., Naya, F., Jorge, J. and Conesa, J.: CIGRO: A Minimal Instruction Set Calligraphic Interface for Sketch-Based Modeling. Lecture Notes in Computer Science, **2669**, (2003) 549-558
20. Wang, C.C.L. and Yuen M.M.F.: Freeform Extrusion by Sketched Input. Computers & Graphics **27**, (2003) 255–263
21. Baohua, S., Mingxi, T., Frazer, J. H. and Haicheng, Y.: Stroke-based Intelligent Sketching Interface”. Proceeding of the 5th Asia Pacific Conference on Computer Human Interaction APCHI2002, (2002) 500-509