Artificial Intelligence for Engineering Design, Analysis and Manufacturing (2002), *16*, 149–171. Printed in the USA.
Copyright © 2002 Cambridge University Press 0890-0604/02 $12.50
DOI: 10.1017/S0890060402163037

Drawing marks, acts, and reacts: Toward a computational sketching interface for architectural design

ELLEN YI-LUEN DO

Design Machine Group, Department of Architecture, University of Washington, Seattle, Washington 98195-5720, USA

(Received September 10, 2001; Accepted April 8, 2002)

Abstract

Architects use sketching and diagramming in their design process to perform functional reasoning, formal arrangements, analogy transfer, structure mapping, and knowledge acquisition. This paper describes a research framework of the author's efforts in the studies of design drawings and the building of computational sketching tools to support the early conceptual design process in architecture. The first part of the paper discusses empirical studies conducted to determine or guess a designer's thought process from sketches and thus identifies domain-specific graphical symbols. It proposes a reasoning process framework of drawing marks, acts, and reacts. The second part of the paper illustrates how design support tools could be developed based on these concepts and describes the various applications of the study, such as indexing and retrieving of design drawings or images based on the recognition of geometric shapes and the spatial relationships among them.

Keywords: Design Computing; Sketching Interface; Architectural Drawing; Design Reasoning; Diagram Symbols

1. INTRODUCTION

Sketching and diagramming constitute an important procedure in the early conceptual design phases in architectural design. Architects are trained to use pencil and paper to develop design concepts and to communicate their thinking through the act of drawing. Design thinking concerns both form and function. It is a form of visual and spatial reasoning. The process of interactively making a drawing (the act of drawing) supports a designer's abilities to visualize design ideas and to test and reason about shapes and functions (react to the drawings). These drawings, or imaginative design artifacts in a tangible form (graphic marks on paper), are then developed further by making more drawings, ultimately resulting in construction.

As computational devices become more pervasive, powerful, and embedded, people are becoming dissatisfied with the standard graphical user interface (GUI) or windows, icons, menus, and pointers (WIMP) interfaces that have dominated computer aided design (CAD) through the 1980s and 1990s. Although these interfaces are improvements over text-only interfaces, they do not support the natural, flexible interactions needed to perform complex tasks (e.g., sketching to perform design). Researchers in many domains are working to enable more natural forms of interaction in a variety of modalities such as natural language understanding, gesture, and handwriting recognition. The trend is toward more human-centric interface design, so that people can interact with computers naturally instead of adapting to machine-oriented procedures.

As a trained architect with design practice experience, I was disappointed and frustrated when (in the early 1990s) I learned many supposedly CAD applications. None of these "design tools" support the most common activities in the early stages of design: freehand drawing and sketching. I then had the opportunity to be involved in building a case-based reasoning design aid called Archie (Domeshek & Kolodner, 1992; Kolodner, 1993; Zimring et al., 1995) and in testing the system's usability (Do et al., 1994). It became clear that case libraries and knowledge-based systems could become more accessible to designers if they had an easy to use interface such as sketching. This revelation started the implementation of the Diagram-Archie program (Gross et al., 1994). It also started a journey of personal investigation

Reprint requests to: Ellen Yi-Luen Do, 208 Gould, Design Machine Group, Box 355720, University of Washington, Seattle, Washington 98195-5720, USA. E-mail: ellendo@u.washington.edu

https://doi.org/10.1017/S0890060402163037 Published online by Cambridge University Press

and research on design intentions in drawings and on the implementation of computational sketching tools to support design. This paper describes the framework of these efforts through an overview of my previous and current research.

How can one build computational tools to support design reasoning by drawing? If we consider drawing as an iterative process of external representation, we can then analyze the component acts of drawing and build software to support these acts. What then are the purposes of drawing, and can we capture the dynamics of drawings computationally? Furthermore, what kinds of "process feedback" occurs in the design drawing process?

In the following sections, we discuss issues and concerns of the role of visual and spatial reasoning in sketching and the implications for models of design thinking and computational tools. Section 2 reviews the critical issues of building computational support for design reasoning through empirical studies of drawing.

We then draw attention to three important aspects of drawing, from the static to the dynamic—drawing marks, drawing acts, and drawing reacts—and describe their implications and corresponding computational tools in the sections that follow. Under the rubric of drawing marks, acts, and reacts, Section 4 describes several prototype systems that we built to support different aspects of design reasoning. Finally we discuss these investigations and the future research directions they suggest.

2. RELATED WORK

This section responds to three questions: What is the role of freehand sketching and diagramming in design? How can one study the reasoning processes of designers so as to further our understanding of sketching in design? What computer programs have already been built to support sketching activities? The first part of this section looks at design studies that focus on the importance of drawing in design. The second part describes cognitive science and protocol analysis studies of the relationship between drawing and design thinking. Finally, the last part reviews relevant computational sketching programs. This section aims to establish a knowledge base and rationale for the empirical studies described in Section 3 and the system building described in Section 4.

2.1. Design studies

Drawing plays an important role in architectural design. Designers use the act of drawing to help discover and explore ideas. They draw to think about design and to remind themselves of possible design alternatives. Although drawing styles may vary, many designers acknowledge the use of sketches as an integral part of their design process.

2.1.1. Studies based on interviews and portfolio reviews

Several recent design studies focus on the connection between design and drawing. Lawson's *Design in Mind* presents interviews with 10 famous architects and analyzes their design approaches in practice (Lawson, 1994). These designers argued that drawing is crucial in their design process because drawing is a "discovering journey of ideas," a "dialogue" (p. 26), and "communication with self" (p. 98). Lawson concludes that these designers "find it hard to think without a pencil in their hand" (p. 141). Fraser and Henmi's *Envisioning Architecture* looks at how techniques used to make different drawing types influence the making of architecture (Fraser & Henmi, 1994). They note that architects "symbolize . . . intangible factors such as movement, access, sound, view, function, and time" (p. 110) in diagrammatic form to represent the abstraction and reduction of information. Herbert's *Architectural Study Drawings* (Herbert, 1993) argues that drawings are "the designer's principal means of thinking" (p. 1). Robbins' *Why Architects Draw*, like Herbert's book, examines the work of well-known professional architects (Robbins, 1994) and quotes architects who argue that sketching "provoke(s) a change of ideas" (p. 153) and helps "verify" decisions to see if they work (p. 157).

2.1.2. Designers' self-reports

Many architects express the importance of diagrams and drawing in their design process. They describe their experiences using drawings to think about design, to explore and record ideas, and to communicate with others. For example, Graves explains that his "referential sketch" serves the purpose of a "diary," or record of his observations and discoveries (Graves, 1977). He also says that he and his partners usually collaborate through a "conversation" by exchanging and adding to the drawings and that this exchange is a game of idea exploration facilitated through a common understanding of a set of explicit "principles or conventions." Louis Kahn in "The Value and Aim in Sketching," mentions that sketches are as important to him as design problems (Kahn, 1931). He explains that "drawing is a mode of representation" (p. 10). Regardless of the medium used, the value of a drawing is in the "purpose" of making. He argues that designers need to interact and work with a sketch, not simply "crystallize" thoughts on paper. Peter Eisenman's *House of Cards* documents his use of diagrams and drawings for a series of house design projects; he calls it a "record of evolution" of his design thinking (Eisenman, 1987).

2.2. Drawing as external representation

Design drawing is an iterative and interactive act involving recording ideas, recognizing functions and meanings in the

https://doi.org/10.1017/S0890060402163037 Published online by Cambridge University Press

drawings, and finding new forms and adapting them into the design. In order to implement desirable functionality in a computational drawing environment, we must first carefully examine how designers use drawing in their design process.

2.2.1. Protocol analysis of design activities

Design studies researchers and cognitive scientists have conducted research on design drawings. One common approach is recording video and verbal transcriptions of design sessions; this is the "think-aloud" protocol method. Eastman (1968) and Akin (Akin, 1978, 1986) use protocol analysis to study design problem solving and the solution generation process. Eastman views the design of a physical environment as a problem-solving task in an information process model, following Newell and Simon (1963). Suwa and Tversky report that architectural drawing facilitates problem solving and creative effort (Suwa & Tversky, 1996). They argue that seeing drawing marks helps architects to refine their design ideas.

Design researchers (Dorst & Cross, 1995; Cross et al., 1996; McFadzean et al., 1999) also employ design protocols as research methods to study designers in action. Van Sommers (1984) uses empirical studies of graphic production to argue that the act of drawing is a "graphic engine or a production system" (p. 245) that helps people generate concepts. Goel's *Sketches of Thought* argues that drawings are "external symbol systems" that can be manipulated and reasoned with to represent real world artifacts (Goel, 1995) and that graphical representations have certain capacities that nongraphical symbol systems lack, for example, the ability to gracefully represent vagueness and ambiguity.

2.2.2. Retrospective studies of design

Another set of studies use introspective, retrospective, or speculative knowledge instead of think-aloud protocols. Galle and Kovács (1992) argue that an introspection record is more compact than a protocol transcript and more detailed than answers collected in an interview. Suwa and Tversky employed retrospective reports of design sessions (Suwa & Tversky, 1996) to study designers' perceptual processes in observing their own sketches. Porter and Schön's "thought-experiment" (Porter, 1988) used a speculative anecdote of a design process to account for the underlying logic of designing. Porter argues that retrospective analysis is a form of inquiry common to design teaching and has implications for the design of computer-based design tools.

2.2.3. Models of design reasoning

Design drawing is also regarded as a means of transformation for design objects. For example, Schön argues that design reasoning is a thinking pattern that uses design rules (Schön, 1988) and a process of "reflection-in-action" (Schön, 1985). He argues that designers first "see," and then "move," design objects (Schön & Wiggins, 1992). Goldschmidt sharpens this notion and argues that design reasoning consists of "seeing as" and "seeing that" modalities (Goldschmidt, 1989). She considers sketching an operation of design moves and arguments, an "oscillation of arguments" that brings about a gradual transformation of images (Goldschmidt, 1991). Ullman, Wood, and Craig argue that in a design each marking action is an external representation of a chunk of information (Ullman et al., 1990).

2.3. Computer systems with sketching interfaces

Several researchers have used the term sketch in naming their design decision support systems. They generally use the word sketch to advocate the idea of drawing with a pen or having an easy to use interface, although, in many cases, their systems only support hard-line drawing. (Strictly speaking, these hard-line drawing, or palette input, systems should not be called sketch systems.) However, many interesting ideas, such as using constraints in a drawing environment (e.g., Sketchpad, Sutherland, 1963; interpreting sketches as straight lines, e.g., STRAIT, Taggart, 1975; SketchIT, Stahovich, 1996; SKETCH, Zeleznik et al., 1996) have been explored in these system-building efforts. The following section discusses several systems that either have sketch in their name, convert sketches into straight lines, or support freehand drawing input.

2.3.1. Systems with sketch in their name

Ivan Sutherland's Sketchpad (1963) developed several important interface concepts that are common in today's computer systems (e.g., constraints, copying and editing, grouping). It enabled users to draw primitive graphic objects with a light pen, and thus provided a new human–machine interface by eliminating typed commands "in favor of line drawings" (Sutherland, 1963). Zeleznik et al.'s (1996) more recent SKETCH project explored the idea of sketching gestures as an interface for 3-dimensional (3-D) geometric modeling. For example, to construct a rectangular solid, the user clicks and drags three lines along the *x*, *y*, and *z* axes to specify its dimensions. Stahovich's SketchIT program (1996) reads a mechanical sketch and transforms it into multiple interpretations of workable designs in the domain of springs and kinematics joints. The sketcher in the SketchIT project does not support freehand sketches. Instead, it provides a tool palette with objects such as face, pivot, and slider. All these systems simply support structured drawing. The drawings are presented as well-defined rectified objects: no sketchy lines, either as input or output, are involved.

2.3.2. Systems that convert sketches to objects

Many sketch systems take sketchy, rough drawings as input and convert them into straightened objects or clean curves. For example, Taggart's STRAIT converts sketchy lines into intended straight lines and sketchy curves into

https://doi.org/10.1017/S0890060402163037 Published online by Cambridge University Press

polylines (Taggart, 1975). He argues that sketch recognition is the way to "communicate architectural intentions and ideas" with a computer because designers will enjoy the comfort of a "familiar (pencil) tool" instead of the "input protocol" demanded by the systems. Sketch-Solid (Sivaloganathan, 1991) turns sketchy lines on an isometric grid into 3-D objects. Baudel argues that designers and artists prefer to redraw (rather than to edit) sketches to modify them, and therefore he proposed a "mark-based interaction" technique for editing curves by tracing over them with freehand drawings to indicate more precisely the desired curvature (Baudel, 1994).

2.3.3. Sketch systems that support sketching activities

Recent advances in digitizer and pen computing technologies have induced a renewed interest in research in computational environments for freehand sketching. These systems used pen-based interfaces to input and edit design drawings. Unlike the sketch projects described previously, which use the word sketch to mean an easier interface for drawing or the ability to translate sketches into structured drawings, these systems support the display and manipulation of freehand sketches as they are entered. For example, PerSketch (Saund & Moran, 1994) allows the user to select portions of a freehand drawing by overtracing. "Translucent patches" (Kramer, 1994, 1995; Genau & Kramer, 1995) maintains handwriting and sketchy objects and supports moving irregular sketched shapes. The SILK (Landay & Myers, 1995; Landay, 1996) system interprets freehand drawn objects as interface objects and allows the user to interact with the objects (e.g., move the slider, push a button, etc.). The nuSketch system (Forbus et al., 2001) allows the user to mix drop-in object symbols (bridge, terrain, etc) for a military course of action sketch with hand-drawn sketches. However, the system does not attempt to perform shape recognition of the sketches. Rather, it depends on voice input and specific selection procedures from the user to define object types and names. The only recognition nuSketch handles is the user gestures of arrows and lines to indicate placement and dimension. The EsQUIsE system (Leclercq, 2001) recognizes characters in space labeling, and the lines in schematic design are inferred as boundary definitions. It then uses the information to deduce the characteristics of the rooms being designed. For example, by recognizing space names as contained within partition lines, the system can generate the square area of the rooms and an energy consumption estimate.

3. DESIGN DRAWING STUDIES: VISUAL LANGUAGE IN DESIGN

Sketching in design covers a wide variety of activities that take place in different settings. In order to build useful computational sketching tools for design, we must identify the dimensions of sketching. This includes identifying drawing components and design entities and the spatial relationships and transformations among them.

3.1. Domain-specific graphical symbols

To what extent is it possible to infer, interpret, or even guess what a designer was thinking about by looking at the drawing that was made? Can one find diagramming conventions among designers for functional reasoning tasks in design, such as space planning and furniture arrangement? To answer these questions we conducted several empirical studies on design and drawing. The studies include data analysis of 62 architecture students' concept diagramming and video transcripts and protocol analysis of four architects performing an architect's office design.

3.1.1. Diagramming experiment and protocol analysis of design

The first study (Do, 1995) used design stories and diagrams from a case-based design aid called Archie (Kolodner, 1991; Domeshek & Kolodner, 1992; Zimring et al., 1995). Archie's database contains stories, problems and responses from postoccupancy evaluation data collected in field studies of 10 courthouses and libraries. All related items are cross-linked. Each participant was given tasks of drawing diagrams from stories, writing stories from given diagrams, pairing diagrams and stories, and commenting on given Archie diagram–story pairs.

The second study (Do, 1998) involved the protocol analysis of four designers in action. The designers were given a program brief of an architect's office space design and asked to focus on four different concerns in conceptual, schematic design. The tasks included spatial arrangement, lighting, visibility and privacy, and fitting a special piece of furniture into the design.

3.1.2. Graphic convention: Primitives and symbol configuration

From the diagramming experiment and the design protocol analysis, we found that designers use graphic symbols to represent certain physical objects in design tasks and concerns. For example, when thinking about spatial arrangement, designers would draw geometric primitives (e.g., circles and boxes) to represent different functional spaces with text labels [Fig. 1(a)]. When thinking about objects' (desks, chairs, and door) placements in a room, designers would draw graphic symbols for furniture [Fig. 1(b)]. When thinking about lighting concerns, designers would draw a configuration consisting of an arrow intersecting a vertical line, representing a light ray in a sectional view [Fig. 1(c)].

These graphic symbols are drawing conventions used in the domain of architecture to represent conceptual entities and graphic relationships. The participants from the experiments chose primitives from a limited universe of geometric shapes and symbols to draw their diagrams and composed

https://doi.org/10.1017/S0890060402163037 Published online by Cambridge University Press

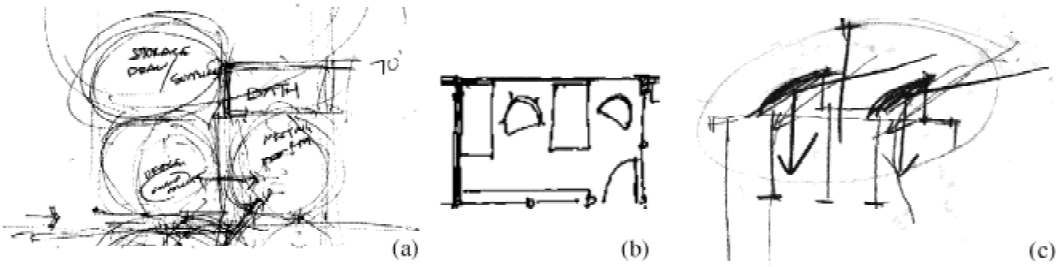


Fig. 1. (a) A bubble diagram for spatial arrangement, (b) an office layout with furniture symbols, and (c) the lighting concerns in a roof section.

them in highly conventional ways. They predominantly used lines, ovals and blobs, rectangles, and hatching. The basic drawing elements—lines, arrows, and geometric shapes—are called "primitives" (Fig. 2).

Primitives are used in diagrams in a variety of different domains, and they are not limited to architectural design. Primitive drawing elements used in diagrams include arrows, lines, hatches and simple geometric shapes (Fig. 3). When primitives are combined, they can form symbols to represent design entities, or domain-specific architectural objects, such as walls and windows, or to illustrate natural phenomena, such as the sun and human figures. For example, a circle was composed with radial lines to indicate the sun; lines were composed to indicate walls and windows; and a circle with lines or a blob was drawn to indicate people (see Fig. 4).

The observation that these designers only used a limited set of graphic symbols suggests that a computer-based diagramming tool would only need to provide limited pattern recognition facilities. These graphic symbols are the visual language for design reasoning (and should be the basis for computer sketching, discussed later in Section 4). They consist of the following vocabulary and syntax: basic geometric shapes, or graphic primitives (lines, circles, boxes); drawing conventions for objects (furniture, building elements, label); drawing context and intentions (drawing view, spatial configurations); and spatial relations (concentric, above, intersect) among drawing elements.

3.2. Overtracing, repetition, and dimensional reasoning

Besides the drawing conventions designers share, the design protocol further revealed several interesting features in the process of making drawings. First, designers use overtracing and redrawing, either directly on the drawing or on another trace layer, to add and refine details. Second, repetitive tasks (such as assigning chairs for each cubicle) are often performed in a sequence. Third, when the task calls for functional reasoning, a designer often draws dimensional markers and makes calculations. Fourth, designers would draw 3-D (perspective or isometric) views to visualize a 2-dimensional (2-D) shape into 3-D space. Next, I describe these observations individually.

3.2.1. Overtracing and redrawing

A common character of designer's working sketches is overtracing, in which the designer's pen repeatedly outlines a particular shape or area of drawing. This overtracing, or redrawing, serves several functions: selection of a design entity or drawing attention to the element; attending to one or another shape interpretation; and shape refinement, or adding detail to an abstract or roughed-out shape.

Figure 5 shows three designers' use of overtracing in the design process. The first designer circled the lower left portion on the original floor plan twice [Fig. 5(a), left] to focus his attention when considering the area for display (as an-

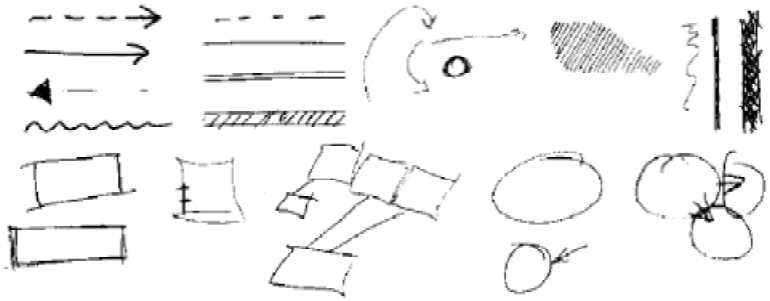


Fig. 2. The primitives (drawing elements) used in diagrams included arrows, lines, hatches, and simple geometric shapes.

https://doi.org/10.1017/S0890060402163037 Published online by Cambridge University Press

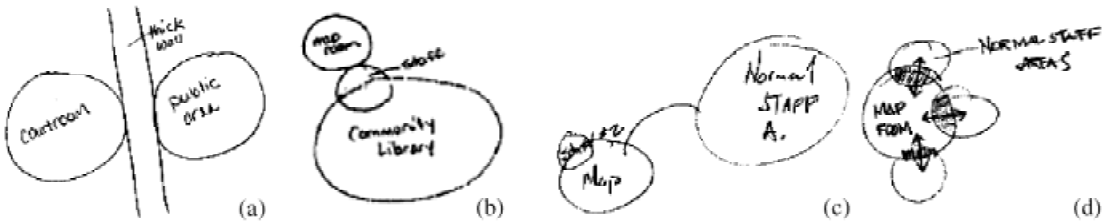


Fig. 3. Diagram examples showing concerns about spatial relations: (a) a thick wall between the courtroom and public area and (b–d) staff control between the map room and community library drawn by three different designers.

notated on the drawing). He drew many overlapping lines on the boundary walls and the inside partitions to test alternatives. He then redrew another floor plan [Fig. 5(a), right] with different space partitions and continued to work on it with many overtraced lines. The second designer first drew a floor plan in vertical orientation [shown as horizontal after rotation; Fig. 5(b) left] and overtraced several partition walls while planning the locations of the lobby and common area. Seeing that the work area should be of main concern, the designer decided to change the perspective. He rotated the paper counterclockwise 90° and redrew a new plan in vertical orientation [Fig. 5(b), right], positioning the working area on top of the plan. He overtraced that newly formed square shape, and called out the wall line to block direct visual access from entry, and wrote down text to explain and record the rationale for the space arrangement of the whole plan.

The drawing technique of repetitive tracing is used to identify aspects of concern, bring focus to the drawn shapes, or specify shape modification. This redrawing and overtracing can happen on one piece of paper or on different trace layers. Figure 6 shows how a designer used two trace layers to facilitate testing design alternatives. The original drawing [Fig. 6(a), left] has a complete spatial layout of the space. Seeing that the chief architect's office appeared to be too small, the designer used this drawing as an underlay and redrew and traced over it on a new layer to make the office bigger [Fig. 6(a)]. He first used the same layer to trace over the left wall of the kitchenette. He then moved the top layer to the right so that this wall lined up with the left side of the underlay conference room and traced the right wall of the conference room. Finally, he moved the trace layer to the left and defined the new location of the kitchenette (right to the conference room) by tracing the right wall of the underlay kitchenette [Fig. 6(b)]. The second layer served as the medium for working out design alternatives. The designer did not redraw the whole floor plan completely.

Figure 7 shows a similar scenario of the drawing and interpreting activities in a design process. The floor plan in the middle [Fig. 7(a)] is the first proposal for the spatial layout. Having identified the service area on the top of the drawing, the designer moved to consider the work area and started a new drawing to its left [Fig. 7(b)]. After partitioning arrangements (with repetitive straight lines), the designer recognized a vertical circulation axis [single line on Fig. 7(b), left]. He redrew the floor plan again [Fig. 7(c)], this time focusing on extending the axial line all the way across the space. He scribbled circular, curvy lines along the axial line to indicate the circulation path while working out the access to the rooms on the right. Then he drew a new and enlarged plan of only the work area [on the right of the original plan; Fig. 7(d)], testing the definition of a corridor (two parallel vertical lines) and a counter or desk space [top left corner of Fig. 7(d)]. He then redrew the whole floor plan [Fig. 7(e)] to work out the reception desk area [bottom of Fig. 7(e)].

This act of drawing, and redrawing can be used to simplify an idea to an abstract form in which no specific shape features are included. Designers also redraw to emphasize a particular area, or to assign a different character to a particular shape. By doing so, the designers see and interpret their drawings to test and evaluate their design decisions. Figure 8 shows a sequence of video clips from the protocol

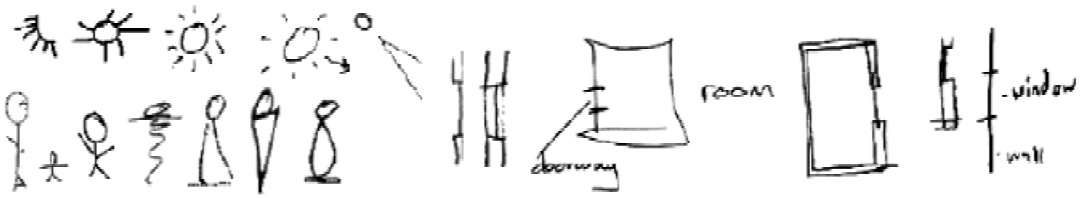


Fig. 4. Primitives were combined to make more complex symbols.

https://doi.org/10.1017/S0890060402163037 Published online by Cambridge University Press

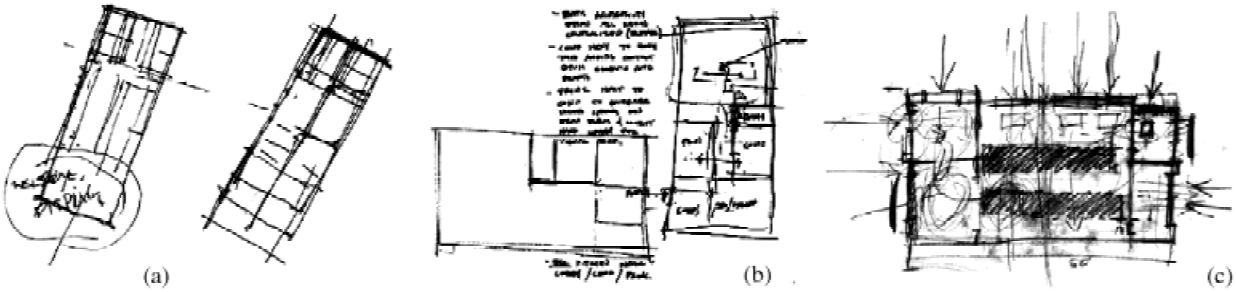


Fig. 5. Overtracing and redrawing to select an object and draw attention to it: (a) consideration of the location for the partition walls, (b) focusing on possible spatial issues of the space, and (c) thinking about how the light enters the building envelope.

analysis in which trace layers are used, not only in plan view, but also sectional (underlay) drawing. The designer used and moved the ruler to check on the dimensions of the various spaces.

3.2.2. Performing repetitive tasks in a sequence

The video protocol of designers in action also revealed an interesting phenomenon not easy to identify in the static drawing marks analysis. In the context of arranging design entities (e.g., multiple work space cubicles, desks, and chairs), a designer would carry out repetitive tasks in sequence. If we look at the run of sequence of drawing acts (in the video clips), we find that the designer had a plan to carry out the tasks in a top-down hierarchical fashion. For example, Figure 9 shows the designer's drawing sequence to produce three identical desk spaces. He started with location registration marks [from left to right; Fig. 9(a)], drew three repetitive vertical lines [Fig. 9(b)] to connect to them to form partition walls, added horizontal tables [Fig. 9(c)], and vertical desks [Fig. 9(d)] in each partition, and finally added the chairs [Fig. 9(e)]. All these objects are drawn from left to right.

The designer would deal with the space first, then place the furniture. He would also place tables before adding chairs. Figure 10 shows another example of duplicate object placement. The designer drew the chairs surrounding the conference table in clockwise direction (five or six chairs). This right-handed designer had a preference to draw in a clockwise direction when placing radial objects. However, this does not mean that the designer would always draw related objects in clockwise sequence. The plan of the sequence of drawing is tied to the object hierarchy: important or dominating objects are drawn first. Figure 11 (middle) shows an arrangement of the work area. The desks in the drawing's bottom part were drawn first, from left to right, and the chairs were added in reverse order, from right to left (Fig. 11,

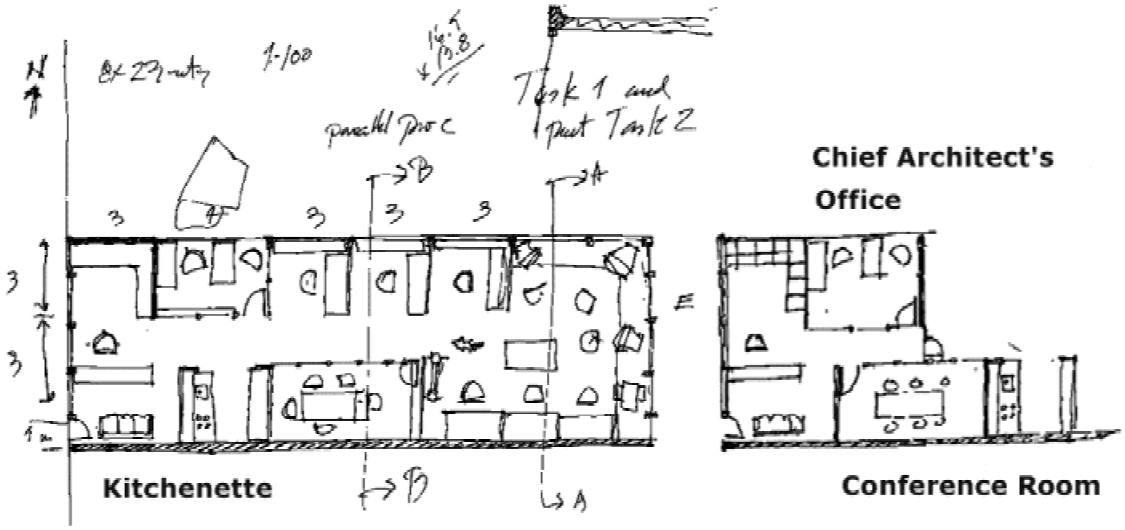


Fig. 6. Redrawing on a different trace layer to test and modify the dimensions of the space (chief architect's office was enlarged) and switching the location of the kitchenette with the conference room. Shown are (a) the original floor plan layout and (b) an alternative layout on a new trace layer (partially drawn with the original floor plan as the underlay).

https://doi.org/10.1017/S0890060402163037 Published online by Cambridge University Press

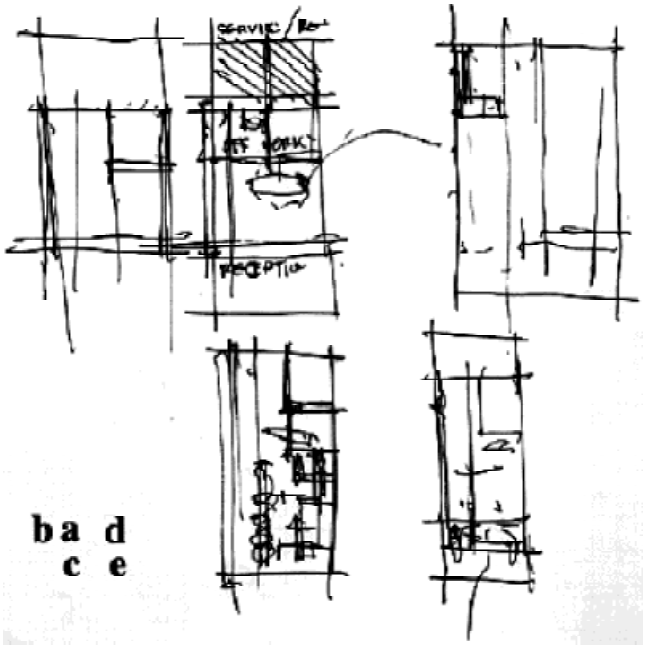


Fig. 7. Redrawing the same shapes to test alternative arrangements (a first, then b, c, d, and e).

left). The top portion of the drawing consists of monitors (drawn clockwise), and then the chairs were added in reverse order (Fig. 11, right).

3.2.3. Functional reasoning with dimensional marks and calculations

The design protocols also revealed the dynamic reasoning process in design. When a design task involved spatial requirements, the designer would not only draw graphic symbols for furniture but also draw dimensional markers and numbers to calculate the size of the space and check requirements. For example, a design task may call for a work space of at least 800 ft^2, which can accommodate a 4×10 ft table in the conference room. Figure 12 shows that the designer drew dimensional markers and labeling and then wrote down numbers and calculations to decide the size and dimension of a functional space. He drew grids and wrote, drew, and calculated the area to make sure his design worked.

3.2.4. Visualization of 3-D geometry

Architectural space is ultimately 3-D. Designers often move or project from 2-D floor plan sketches to 3-D representations such as isometric or perspective views. By doing so, they put themselves in the design context of seeing and testing how the space works in a geometric model that formally expresses the shape of the space. Figure 13 shows a collection of 3-D views drawn adjacent to the floor plan drawings to explore design concerns of lighting and how people and furniture would function in the design arrangements.

3.3. Transformations of design entities

We also performed a retrospective analysis (Do et al., 2000) of a pavilion house design that an architect carried out over the period of 15 years. Unlike a laboratory experiment (e.g., protocol analysis), which usually has a short time span, this project spans a long period of time. Furthermore, this design project focuses mostly on form manipulation rather than the functional problem solving activities that design protocol more often investigates. This study was carried out to analyze the patterns of design operations and relations among drawings. We looked at 110 drawings that the architect selected (and organized for two presentations). Many of these drawings are a composite of several drawing types, such as plan, sections, elevations, and isometric views. Figure 14 shows a sample of these drawings.

The first analysis of the pavilion house design drawings resulted in a diagram to account for connections among the drawings (Fig. 15). We then developed a coding scheme to classify these drawings into different categories. The scheme codes properties of the drawings, such as the elements depicted, as well as projection type, view angles of the building, and element names and locations (Fig. 16).

For any two drawings, the coding scheme also accounts for design element transformations, such as geometric translations of shapes, locations, and color. We use the letter codes E, L, T, and C to indicate respective element identi-



Fig. 8. The sequence of images from the protocol analysis video showing a trace layer of the floor plan overlaid on top of a trace of the section, and a ruler positioned in different locations to measure the dimensions and check if the space will function.

https://doi.org/10.1017/S0890060402163037 Published online by Cambridge University Press

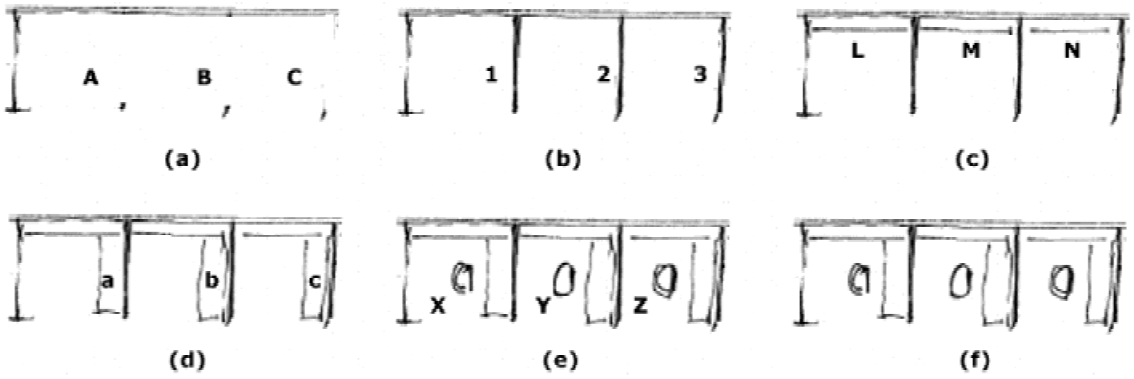


Fig. 9. The drawing sequence (in chronological order, A–B–C or 1–2–3) for the three work space arrangements. (a) Existing wall lines are added via three dots to register locations for the three spaces. (b) Three vertical partition lines are drawn down from the top wall to connect to the registration marks. (c) Three consecutive horizontal lines are added to define the side desks by the wall. (d) Three desks are placed against the partition walls. (e) Three chairs are added. (f) The final space configuration of the three same furniture layouts.

fiers, location identifiers, transformation types, and use of a color; D indicates the projection type of the drawing, V is the view direction, M is the drawing medium, and I is the designer's self-described intention in making the drawing.

For example, the expression below indicates that design element 16 (staircase) at location 4 (middle left) moves down (transformation 4), moves right (transformation 1), and rotates 180 degrees (transformation 17) to location 9 (lower right).

$$E16 \; @ \; L4 \; ((T4 + T17)) \; @ \; L9$$

Figure 17 shows the two drawings, the 3×3 location grid, and the element (staircase) whose transformation is described.

4. COMPUTATIONAL SKETCHING: TOWARD A MORE HUMAN INTERFACE FOR DESIGN

Given these studies and observations about design reasoning and the various ways that drawing seems to support it, we come to an important question. Could artificially intelligent computational design media—"intelligent paper"—support design better, and if so, how? We believe an AI-based sketching program can provide an enhanced environment for design. Drawing can be the interface, or at least an alternative input modality, to various knowledge-based design systems. Our goal is to construct more human-like interfaces for computational design tools. We have implemented several prototype systems to support computational sketching.

At the base level, a computational sketching tool should recognize static drawing marks, such as simple geometric shapes, and their spatial relationships. Designers often use these shapes to perform spatial reasoning and analogy transfer. The second level deals with the act of drawing. Designers engage in the dynamic act of drawing to draw out ideas, to see, to communicate, and to interpret design concerns or perform formal reasoning. Finally, recognizing that it is possible to associate designer's intentions with their drawings, the intelligent paper could be a responsive, or reactive, drawing environment. For example, a simple sketch can be used to access knowledge-based systems such as simulation programs, case libraries, or geometric form making.

The following sections describe some of these efforts under the research framework of drawing marks, drawing acts, and drawing reacts.

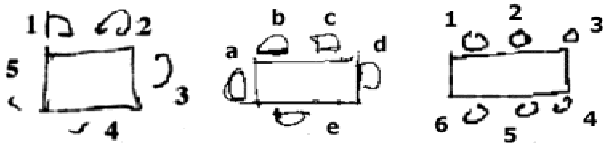


Fig. 10. The drawing sequence (in chronological order, 1–2–3 or A–B–C) for the arrangements of conference chairs surrounding the conference tables (rectangles) in the clockwise direction.

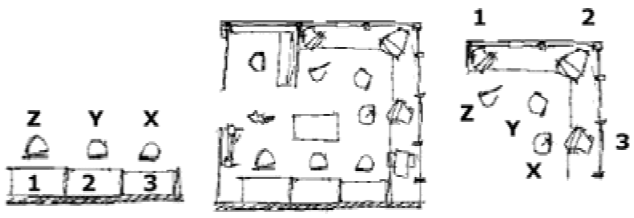


Fig. 11. The drawing sequence (in chronological order, 1–2–3 then X–Y–Z) for the additions of furniture. The original drawing (middle). Three desks (1–2–3) are drawn first then three chair (X–Y–Z) are added in reverse order (left). Three monitors (1–2–3) are drawn first, then their corresponding chairs (X–Y–Z) (right).

https://doi.org/10.1017/S0890060402163037 Published online by Cambridge University Press

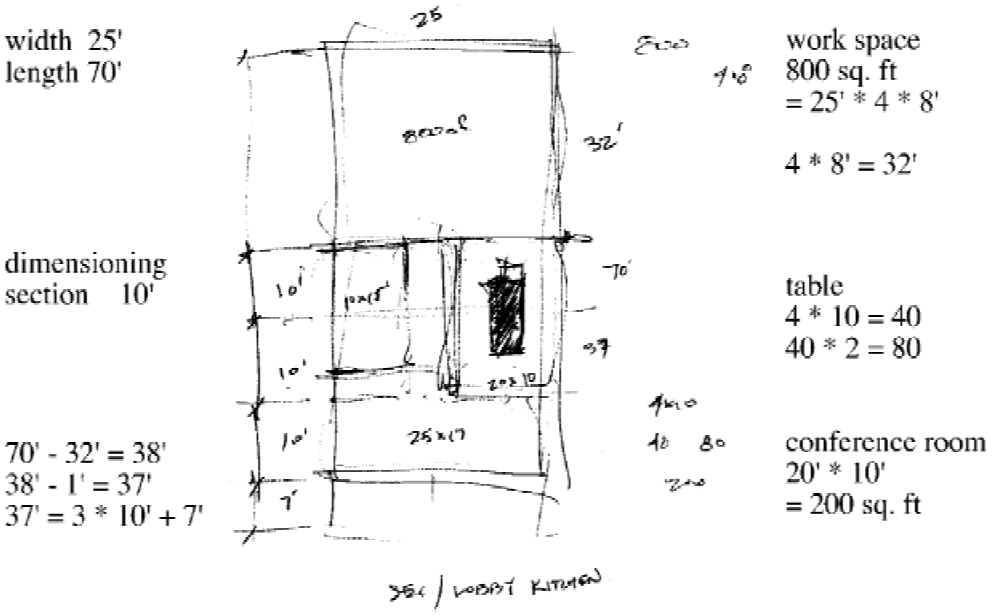


Fig. 12. The dimensional reasoning for the spatial arrangement according to the program square area requirements (numbers, markers, and calculations)

4.1. Drawing marks: Symbol recognition, spatial reasoning engine, and structure mapping

Design drawing is a form of visual language. As described in Section 3, designers share conventions of drawing geometric primitives and symbol configurations. With electronic sketching tools, designers can quickly draw diagrams and sketches to convey symbolic or geometric information. It is important for such tool to take input from the pen (stylus, tablet, LCD screen) and recognize domain specific graphical elements as designers draw them. Unlike paper-based sketching, however, these drawing marks can

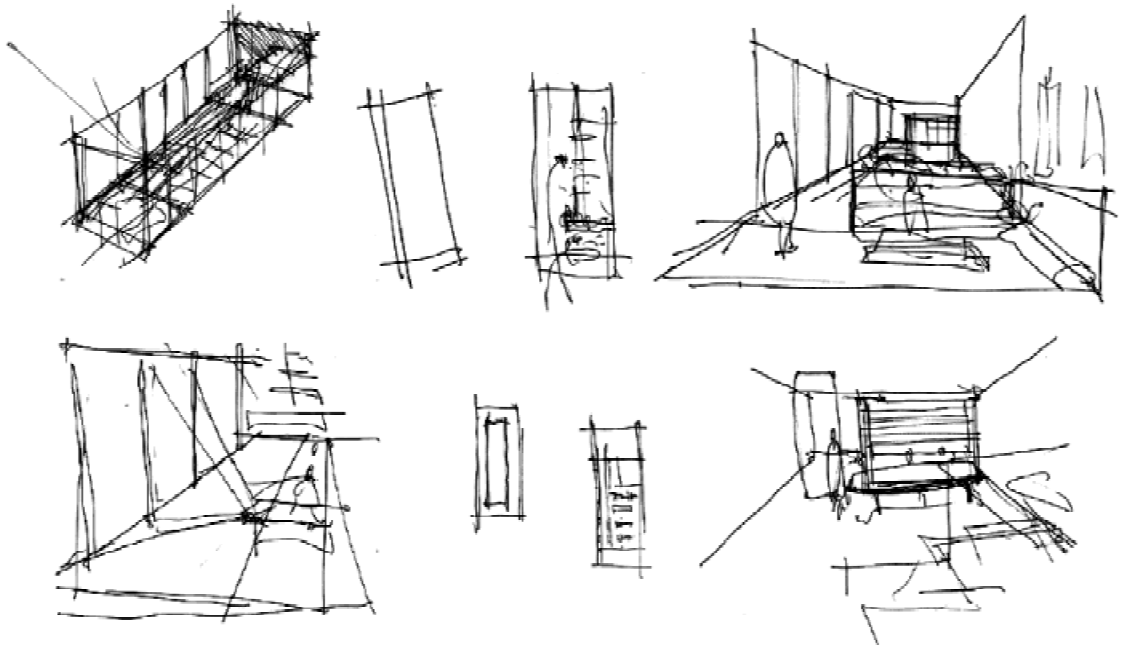


Fig. 13. Three-dimensional views are drawn as isometric or perspective drawings to visualize and test the 2-D floor plan arrangements.

https://doi.org/10.1017/S0890060402163037 Published online by Cambridge University Press

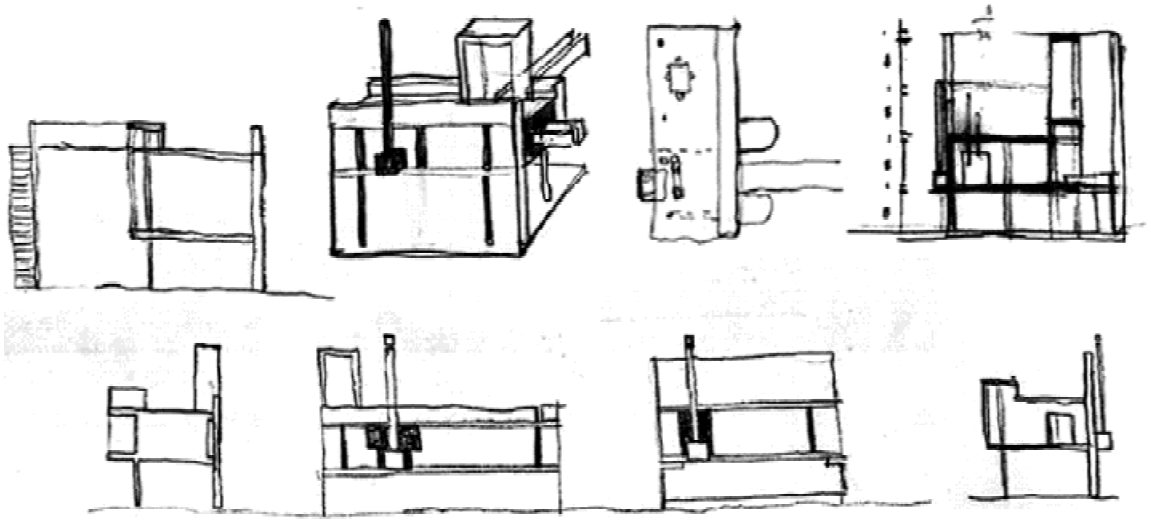


Fig. 14. Examples of a design drawing for the Pavilion House (section, elevation, plan, and 3-D isometric view).

be recorded, recognized, and used to communicate with external applications, such as visual databases.

4.1.1. Recognition of graphic primitives

A computational tool for conceptual design reasoning in architectural should at least be able to perform shape recognition. The Electronic Cocktail Napkin program (Gross, 1996; Gross & Do, 1996) supports recognition and parsing of diagram shapes and configurations. Figure 18 shows that simple geometric shapes called glyphs are recognized through a pen path sequence on a 3×3 grid, the location of corners, number of strokes, direction of the shape, and aspect ratio. The "glyph recognizer" takes initial data input from a pen tablet and parses it through comparison with

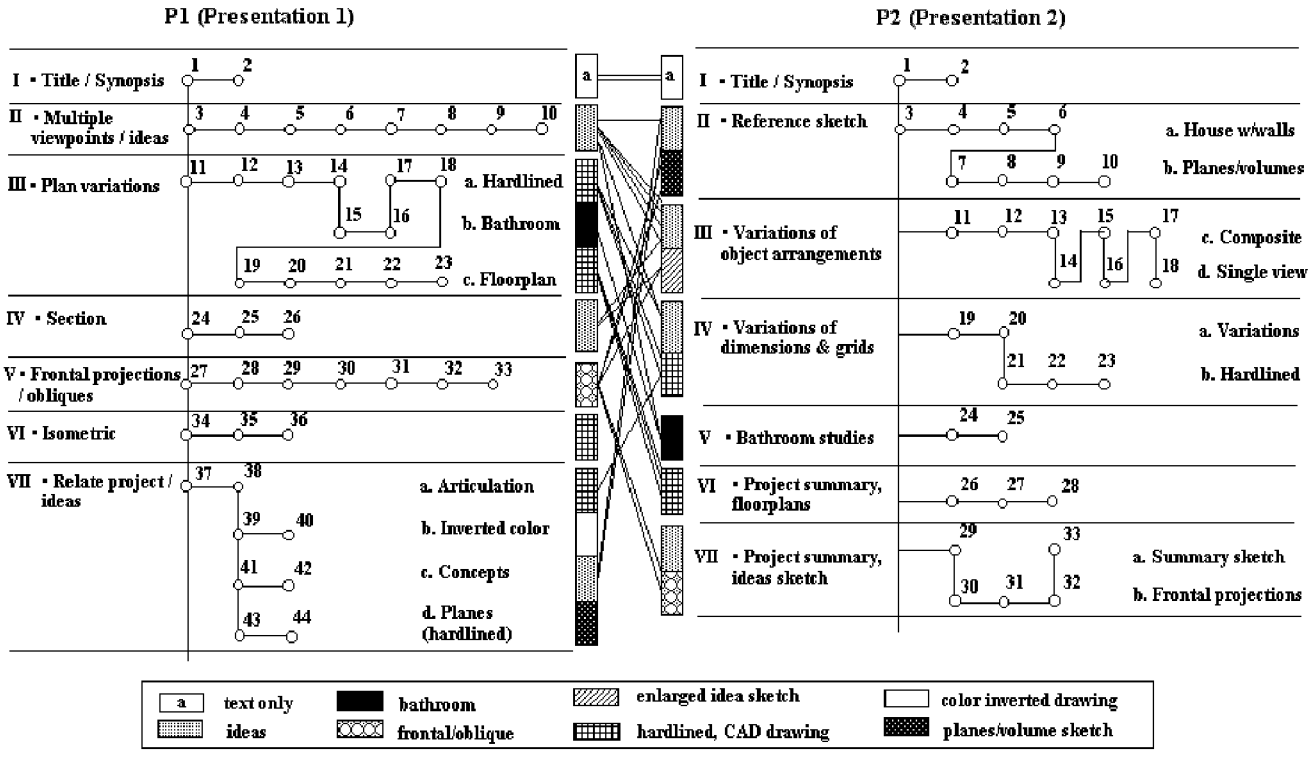


Fig. 15. A network diagram of the relations between drawings.

https://doi.org/10.1017/S0890060402163037 Published online by Cambridge University Press

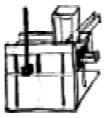
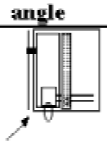
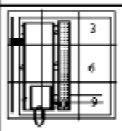
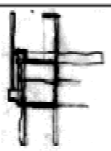
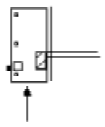
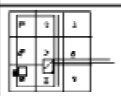
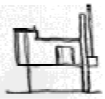
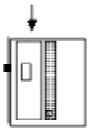
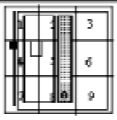
ID #	Drawing	Title	Intention Annotation	Drawing Type	View angle	Elements	Location /scale	Medium
P1-30 (P1-9a) P2-16 (P2-15)		wall & projected volumes (variations on the theme)	isometric front • slots in wall • marking internal grid system on the facade	3D Frontal Isometric (D3+D4)		E1, E2, E3, E4, E5, E6, E7, E8, E9, E10, E12, E12, E12, E12, E13, E14, E15		pencil (M1) yellow, blue, red markers (M4)
P1-6f P2-12f		sectional profile studies	checking spatial relations	section (D3)		E1, E2, E3, E4, E5, E6, E7, E8, E9, E10		pencil (M1) pen (M2)
P1-9f P2-15f		thickened wall and projection	variation of sectional space	section (D2)		E1, E2, E3, E4, E5, E6, E9, E10, E14		pencil (M1)

Fig. 16. Drawings in a coded table of different classifications (type, view angles, location, etc).

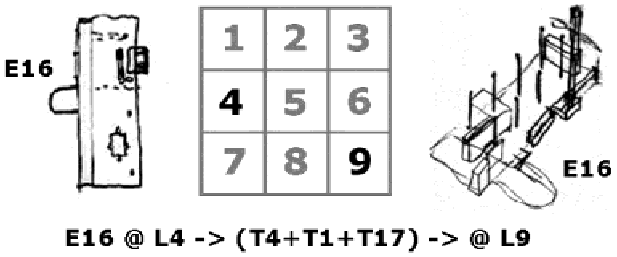


Fig. 17. Element 16 (stair case) in drawing I (left) is moved (T4, down; T1, right; T17, rotate 180°) to a different location (from location 4 to location 9).

stored templates. These templates are learned by the program from the users. Each user can define new symbols by showing the system several examples and naming them.

4.1.2. Spatial reasoning engine

Simple geometric shapes can be combined to form configurations. The spatial reasoning engine in the Electronic Cocktail Napkin program uses graphical rewrite rules to check and produce binary relations between any two objects. This pattern recognizer runs recursively until all conditions are found. These spatial relationships, such as

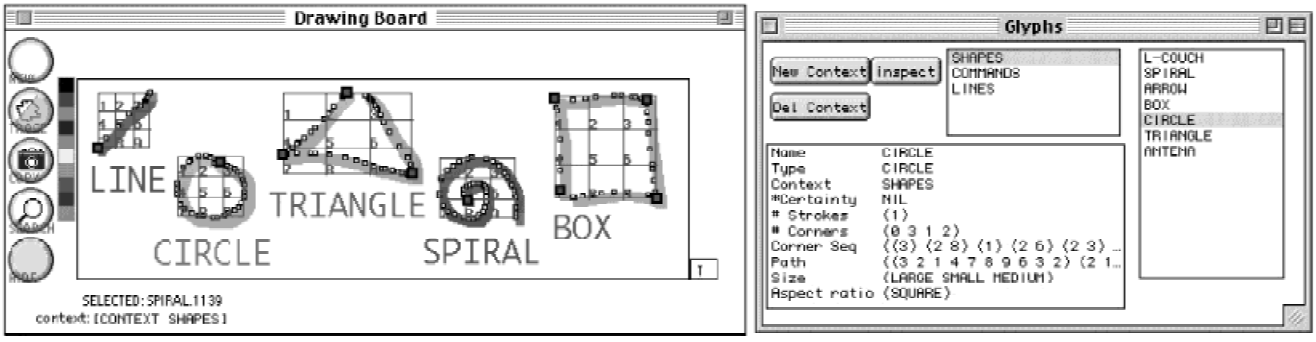


Fig. 18. Simple geometric shapes are recognized with a 3 × 3 grid, drawing sequence, aspect ratio, and identified corners. The drawing board (left) and the glyph recognizer window showing the components of the shape of a circle (right).

https://doi.org/10.1017/S0890060402163037 Published online by Cambridge University Press

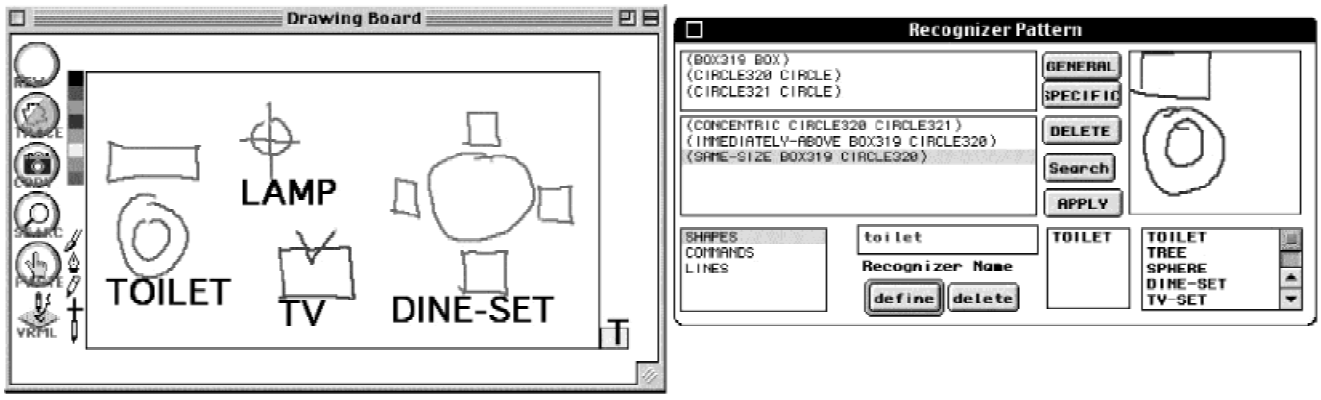


Fig. 19. The configurations are defined by combining geometric shapes and adjusting spatial relationships (concentric, above, same size, etc.) as shown in the configuration recognizer (right).

concentricity, containment, overlap, and intersection, are organized hierarchically and can be relaxed or constrained when user defines a new symbol configuration. For example, Figure 19 shows a collection of user-defined symbols for commonly used architecture elements. A dining table set can be defined as four boxes surrounding a circle. A toilet can be defined as two concentric circles directly adjacent to a rectangle box. The pattern recognizer provides choices of spatial constraints so that a user can, for example, relax a concentric relationship to a more generic type such as containment or overlap.

4.1.3. Structure mapping

With the ability to recognize simple geometric shapes and spatial relationships, computational tools can be built to support diagram indexing and the retrieval of design drawing or images based on similarity. Our drawing analogies and shape-based reminding projects (Do & Gross, 1995*a*, 1995*b*) explore finding visual references based on the adjustable weighting of similarity measures (element types, element counts, and spatial relations). Figure 20 shows the similarity measures of six diagram configurations that have the same element counts. Diagrams with more matching element types and spatial relations among the elements that are like the original query figure (Fig. 20, left) have higher scores.

Figure 21 shows an example of shape-based reminding or query by sketch retrieval of several visual and nonvisual databases. A diagram of a temple elevation retrieves information from a CD-ROM database (Matthews, 1994) of famous architecture, a web site, an image from a photography archive, and a QuickTime movie simulating shadow analysis. In each instance, database items are first indexed with diagrams, and subsequent queries are com-

Figure					
Elements	• Triangle • Box • H-Line	• Half Circle • Box • H-Line	• Triangle • Colonnade • H-Line	• Line • Triangle • Box	• Box • Box • Box
Relations	• Triangle d-above Box • Box d-above H-Line	• Half Circle above Box • Box d-above H-Line	• Triangle d-above Colonnade • Colonnade d-above H-Line	• Line d-above Triangle • Triangle d-above Box	• Box above Box • Box d-above Box
Similarity	Element type	2/3	2/3	1/3	1/3
	Relations	1/2	1	1	1/2
	Relations & Type	1/2	0	0	0

Fig. 20. A similarity measure based on element types and spatial relations. All these figures have the same element count. (The row of lines in the middle diagram is recognized as a colonnade object.)

https://doi.org/10.1017/S0890060402163037 Published online by Cambridge University Press

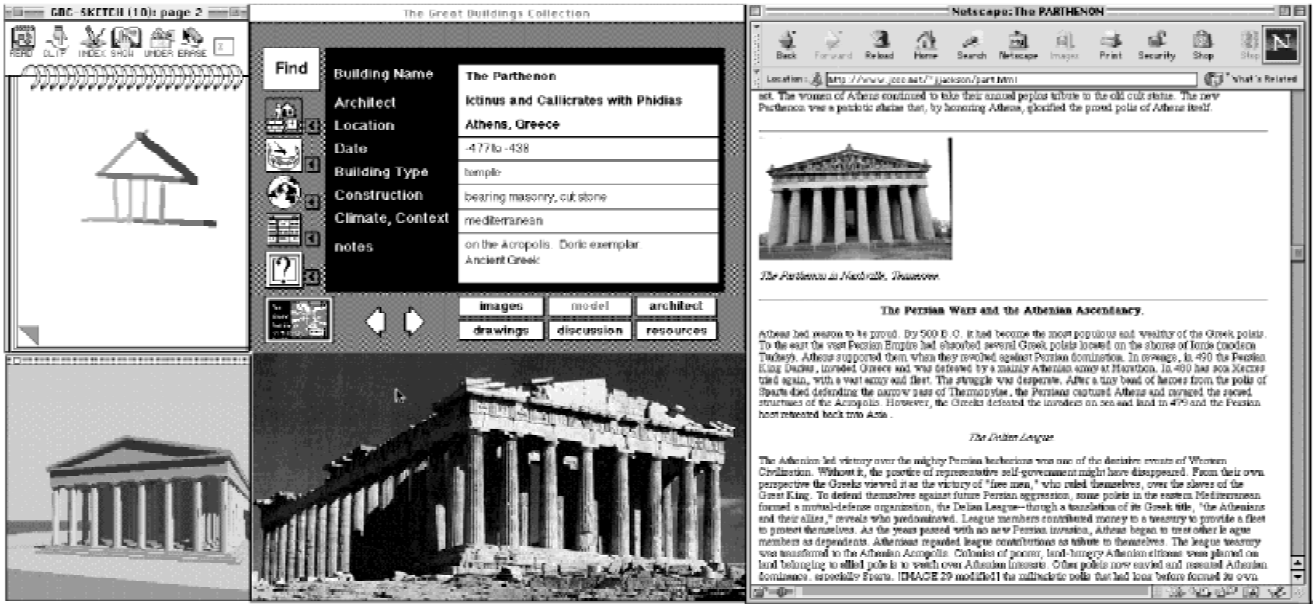


Fig. 21. An example of shape-based reminding: a diagram query retrieves an index card record from the Great Buildings Collection CD ROM (top middle), a Web page about the history of the Parthenon (right), a JPEG image from a photo archive (bottom middle), and a QuickTime movie of shadow analysis (bottom left).

pared with these index diagrams, which are stored in a sketchbook.

4.2. Drawing acts: Layers, filtering, design capture, and transformation

Computational sketching tool should also support designers' active engagements of drawing and capture of the design process. In the design process designers move among different tasks such as concept formation, form making, and reference drawing. These activities are performed through drawing, redrawing, and overtracing on paper and trace layers.

4.2.1. Drawing management, opaque paper, layers, and transparent windows

From the studies of designers in action, we found that designers often use tracing paper to copy parts of drawings, to work out design alternatives, and to move and rotate design elements. Our sketching tool thus incorporates several ways to manage the use of different layers. At the basic level, the drawing board on the Electronic Cocktail Napkin program acts like a piece of opaque paper. However, one can import an image into the drawing board as an underlay and trace over it. For example, the designer can annotate and call out important details and element specifications on the drawing by using a floor plan as an underlay (Fig. 22,

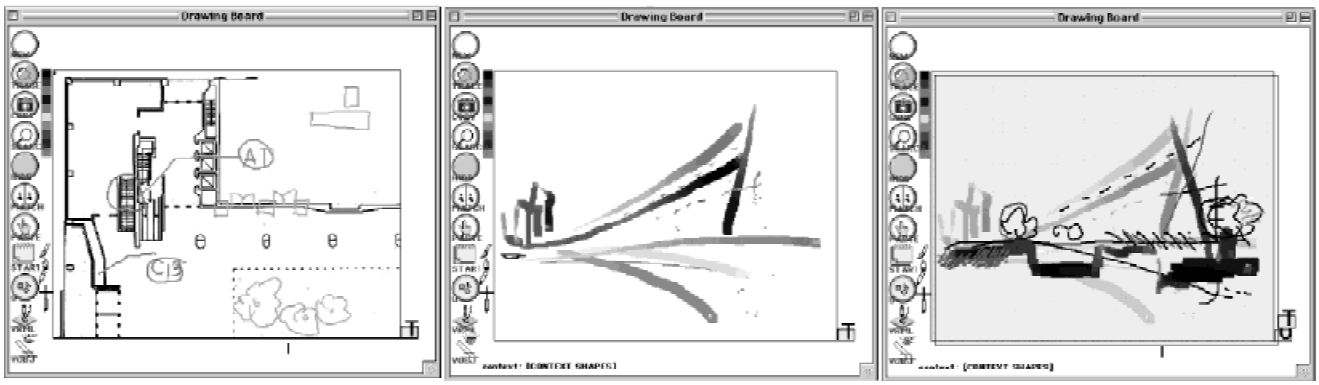


Fig. 22. An underlay floor plan picture is brought into the drawing environment for annotation (left). The designer can add and draw on a new translucent trace layer (right) on top of the drawings on the paperlike opaque base layer (middle).

https://doi.org/10.1017/S0890060402163037 Published online by Cambridge University Press

left). A designer can also put a trace layer on top of the base layer (Fig. 22, middle) to work on design alternatives. Our program renders the new trace layer translucent (Fig. 22, right) so the bottom drawings are visible for reference.

When there are too many trace layers, management among drawings become cumbersome and the lower layers are less legible (each trace layer is partially opaque). Furthermore, designers often work on different tasks and draw references or inspiration from different places. For example, a designer might be working on a spatial layout problem while viewing related documents or image collections from her sketchbook. We implemented a transparent window so that a designer can move the transparent drawing paper over any other application. Figure 23 shows that the designer can sketch on a transparent window that is placed over a movie of a computer simulated walk-through, a CAD model, and a painting from a sketchbook.

4.2.2. Drawing filtering

Besides acting as a means to indicate visual access or a circulation path, drawing also acts as a means to duplicate, highlight, or extract important features from existing imagery, including previously made drawings. A designer often traces over a drawing to draw her attention to the features or details for refinement or the exploration of design alternatives. Given that many overtraced lines are closely positioned, we can provide a filtering function to simplify a sketch or bring out the essential parts. Figure 24 shows two types of filtering provided by our sketching environment. The first simply discards pressure information in the drawing to show its essential form. For example, an initial sketch might deliberately use pen pressure to deliver a watercolor-like effect. Representing the same drawing with a line-only interpretation shows the bare-bones construction of the drawing (Fig. 24, left). Another filtering function reduces overtraced drawings to a simpler form by replacing multiple elements in approximately the same location with a single element. It also eliminates elements smaller than a certain size or those the program was unable to identify (Fig. 24, right).

4.2.3. Design capture

The act of drawing is widely acknowledged to be a means of design exploration, by which the designer can quickly try things out. Researchers often conduct protocol analysis to analyze designers' actions and rationale. Transcribing audio and video information from the protocol analysis data is a time consuming task. It would be valuable to have a tool to record both drawing actions and the verbal think-aloud protocols together for future review. Such a program could also be used for recording conversations among design team members. This is a form of multimodal input. We

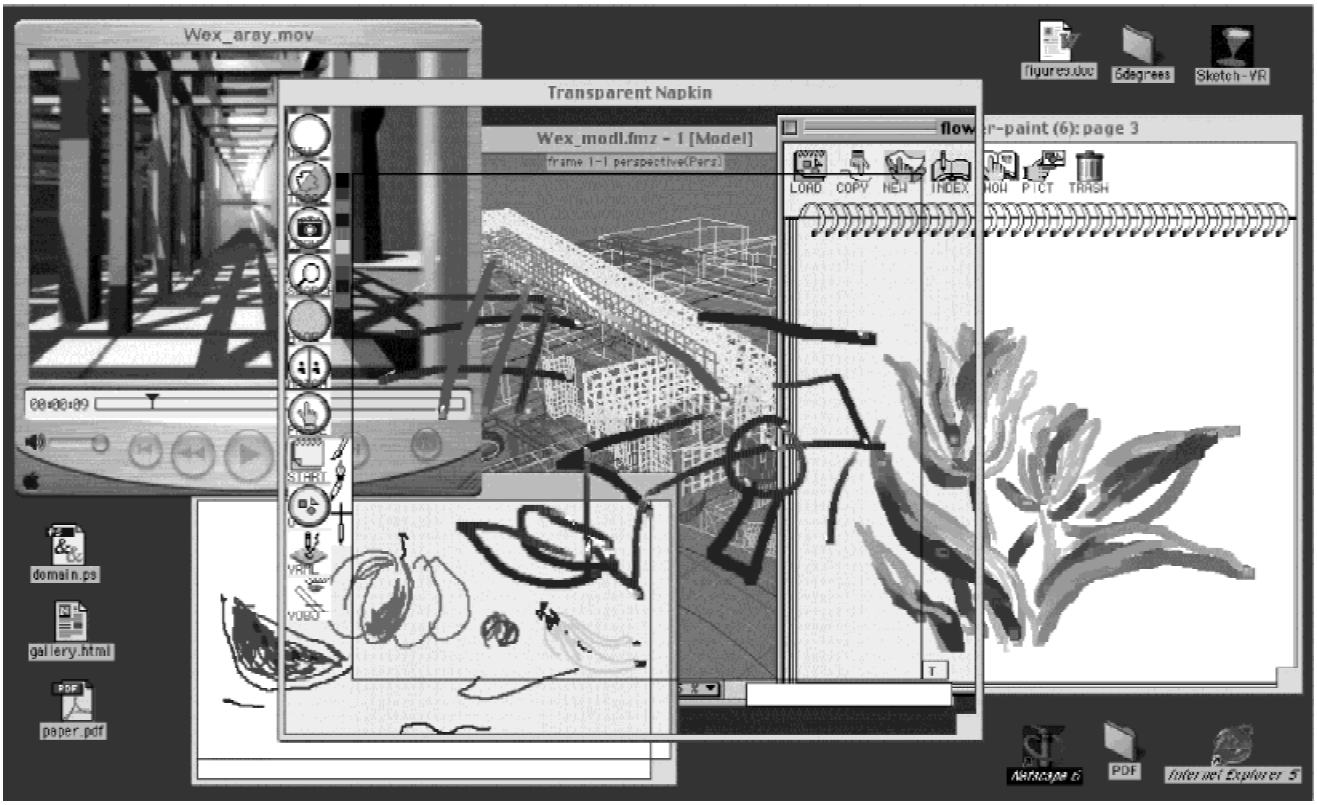


Fig. 23. Transparent window overlays on top of different applications (left to right: a movie, a JPEG image of a sketch, a 3-D CAD model, and a painting from a sketchbook). The designer can sketch and trace images in the transparent window.

https://doi.org/10.1017/S0890060402163037 Published online by Cambridge University Press

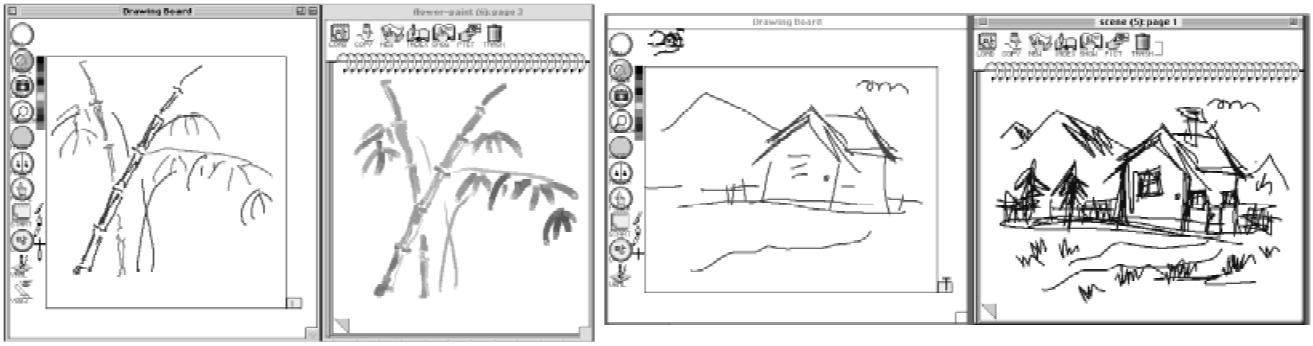


Fig. 24. Filtering of drawings to show the abstract, essential form. A diagram of simple strokes filtered from a watercolor-like image painted with pressure (left). Simplified shapes filtered by eliminating overlapping redraws and small objects (right).

implemented a program called Design Amanuensis (Gross et al., 2001) to explore recording and playing back design histories. The Design Amanuensis consists of both the capture of drawings, using our Cocktail Napkin graphic engine, and the recording and transcription of spoken protocol, using commercial speech recognition software (IBM ViaVoice). Our program tags both verbal and drawing information with time stamps and relates these data so that the whole design drawing process can be reviewed and replayed by selecting either a drawing element or a word in the text of the speech transcript. Figure 25 shows the Design Player, which controls the playback of speech synchronized with the drawing. One can select a drawing or a word to start the playback of the design history. Each word or drawing is highlighted as it is played.

4.2.4. Transformations of design elements

If we compare any two drawings in a design, we can examine the means of arrangement and transformations (design moves) performed between them. To explore and analyze the graphic actions in which design intentions are expressed, we built a prototype system called GIDA, which stands for Graphics Interpreter of Design Actions (Do, 2001*a*). The idea is to create a graphical spreadsheet sorting program to analyze individual drawings as well as the transformations of elements among a series of drawings. This work is an extension and investigation following the coding scheme described in Section 3.3.

The GIDA system allows a researcher to diagram over a picture underlay of a design drawing and to generate analysis of the drawing itself, as well as its relationships with other drawings. For example, the topological and geometric relations among parts of a diagram can be recorded and compared to another diagram traced from a different design drawing in order to reveal the spatial transformations among the elements. Figure 26 (left) shows two drawings (from the Pavilion House study described earlier) selected for comparison. These two drawings are alternatives for the frontal façade design of the same project. In order to record and analyze how element configurations are transformed be-

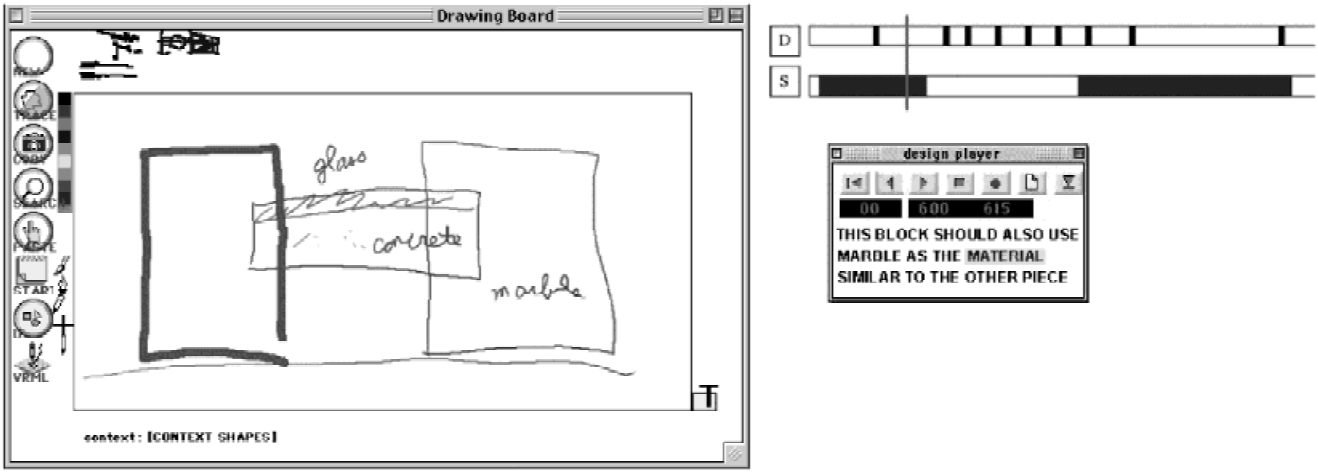


Fig. 25. Design Player (right) plays back part of the design record. The drawing and the speech are synchronized (top right diagram). As each word is highlighted in the voice playback (shown in text dialog), the corresponding drawing is highlighted (left drawing board).

https://doi.org/10.1017/S0890060402163037 Published online by Cambridge University Press

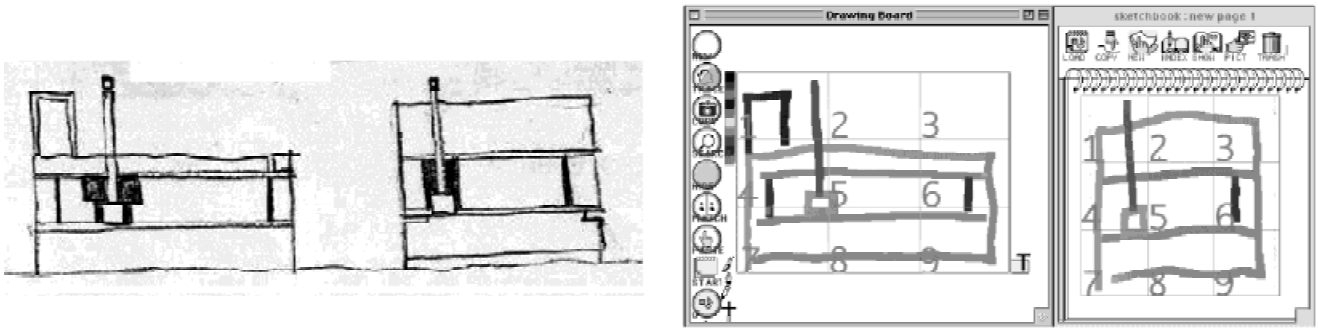


Fig. 26. Two design drawings (left) are used as an underlay to make diagrams of design entities with GIDA's location identifier (3×3 grid) for comparison (right).

tween these two façades, we brought them into GIDA as underlay pictures and drew analytical diagrams on top of them. Figure 26 (right) shows these elevation diagrams traced over the pictures with the underlay removed and a 3×3 location identifier grid overlay. Each design entity is represented as a cell sequence. The resulting list of occupied cell numbers (drawn sequence) for each object is shown in Figure 27.

The comparison table shows that one drawing has eight elements and the other has six. The first drawing has a hood, or canopy-like, element and an extra vertical window that the second drawing does not have. Each element has a position in the global coordinate system and a list describing the local, relative position cell sequence. Upon comparing the lists of the same element from different drawings, the transformation can then be inferred. For example, the Thick Wall's cell sequence list was changed from (7 4 5 6 9 8 7) to (7 4 1 2 3 6 9 8 7). The size (bounding box) of this element in the first drawing is smaller than in the second. Therefore, it is inferred that the transformation for this element from the first drawing to the second one is an enlargement (addition of grid cells 1 2 3). The hood element in the first drawing is removed from the second drawing. Likewise, the transformation of vertical window 2 is a repositioning from (6 9) to (6).

4.3. Drawing reacts: Intention inference, automatic knowledge acquisition, analysis, and testing

Design drawing acts serve different functions for designers. First, designers record mental images onto paper. Second, they see and react to their drawings and perform "seeing as," "seeing that," and then "moving." Designers perform process feedback by interacting with their design drawings. This form of design reasoning is carried out through visual images and spatial manipulations of drawing elements. I argue that, besides recognizing drawing marks and facilitating drawing acts and drawing management, a computational sketching tool should also react to design drawings and provide support from knowledge-based systems. Ultimately, drawing should be the interface with which a designer can interact with design, analysis, and simulation tools.

4.3.1. Intention inference and the right tool at the right time

It is important for a computer system to recognize and support this kind of spatial arrangement and functional reasoning tasks. The Right-Tool-Right-Time system (Do, 1998) is built to identify contexts of design tasks and intentions and use them to activate different knowledge-based design

Design Element	Drawing #1	Drawing #2
E3 Thick Wall	(7 4 5 6 9 8 7)	(7 4 1 2 3 6 9 8 7)
E4 Chimney Box	(7 4 5 8 7)	(7 4 7)
E6 Chimney Pipe	(1 4)	(1 4)
E12 V-Window-1	(4 7)	
E12 V-Window-2	(6 9)	(6)
E7 Hood/Canopy	(4 1 4)	
E13 H-Strip-1	(4 5 6)	(4 5 6)
E13 H-Strip-2	(7 8 9)	(7 8 9)

Fig. 27. A comparison table of two drawings showing a cell sequence list of location identifiers for each element.

https://doi.org/10.1017/S0890060402163037 Published online by Cambridge University Press

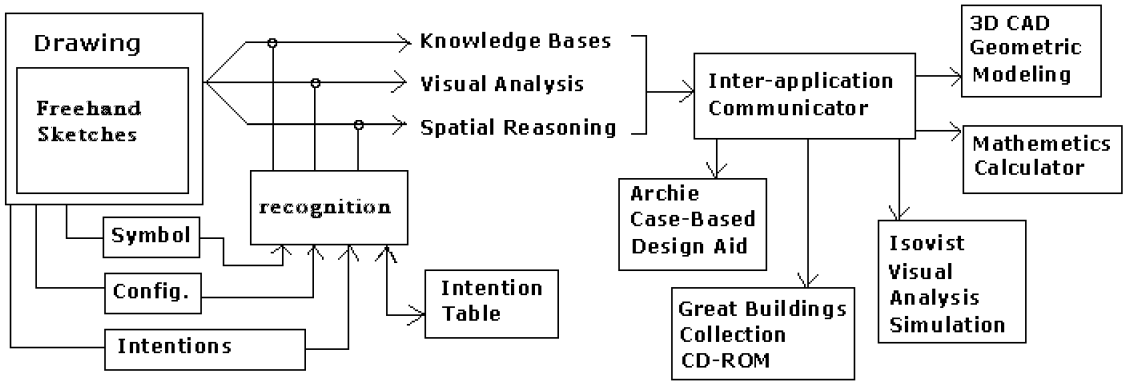


Fig. 28. The Right-Tool-Right-Time program facilitates interactions with design tools through a drawing interface.

tools. For example, recognizing dimensioning markers and calculations will cause a calculator to appear; recognizing sectional drawings with light rays will identify the context and intention as lighting concerns and therefore activate a lighting simulator. Basically, designers interact with the various design tools through the drawing interface. The Right-Tool-Right-Time manager parses the drawing into geometry and configuration and derives the mapping of intentions to corresponding, knowledge-based design tools. Figure 28 shows the architecture for the Right-Tool-Right-Time program.

4.3.2. Knowledge acquisition

Designers often perform design reasoning based on prior experience or design precedence case examples. Case-based design aids such as Archie are built to support this type of reasoning. These knowledge-based design systems often use keywords for indexing and retrieval. For visually oriented, graphically thinking designers (McKim, 1980; Laseau, 1986), a text-based interface may be an obstacle to effective use. This concern was validated through usability studies of the Archie system (Do et al., 1994). Archie's library is filled with postoccupancy evaluation cases containing stories, problems, and responses from postoccupancy evaluation data collected in field studies of about ten courthouses and libraries. All related items are cross-linked. However, to access Archie's design information, one must use specific keywords assigned by the system developer. Designers in the usability study suggested using drawing as an interface to access Archie. Therefore, a prototype of a diagramming tool was developed that helps users to find cases in the Archie database using only hand drawn sketch diagrams (Gross et al., 1994). Figure 29 shows a case story about the arrangement of adults' and children's sections in a library that was retrieved by a diagram in which annotated shapes indicate spaces.

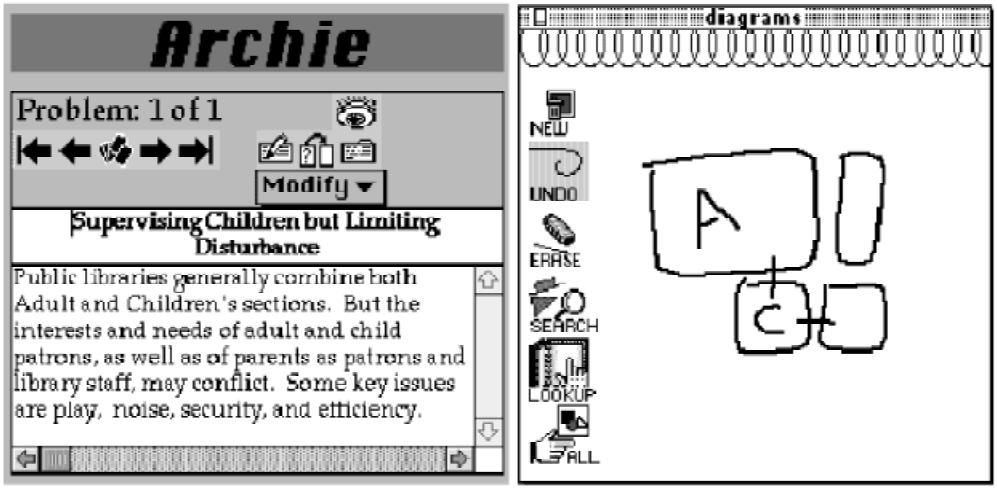


Fig. 29. A story from an Archie case library about the concerns of adult and children's sections arrangement in a library indexed and retrieved by a diagram (A, adult; C, children).

https://doi.org/10.1017/S0890060402163037 Published online by Cambridge University Press

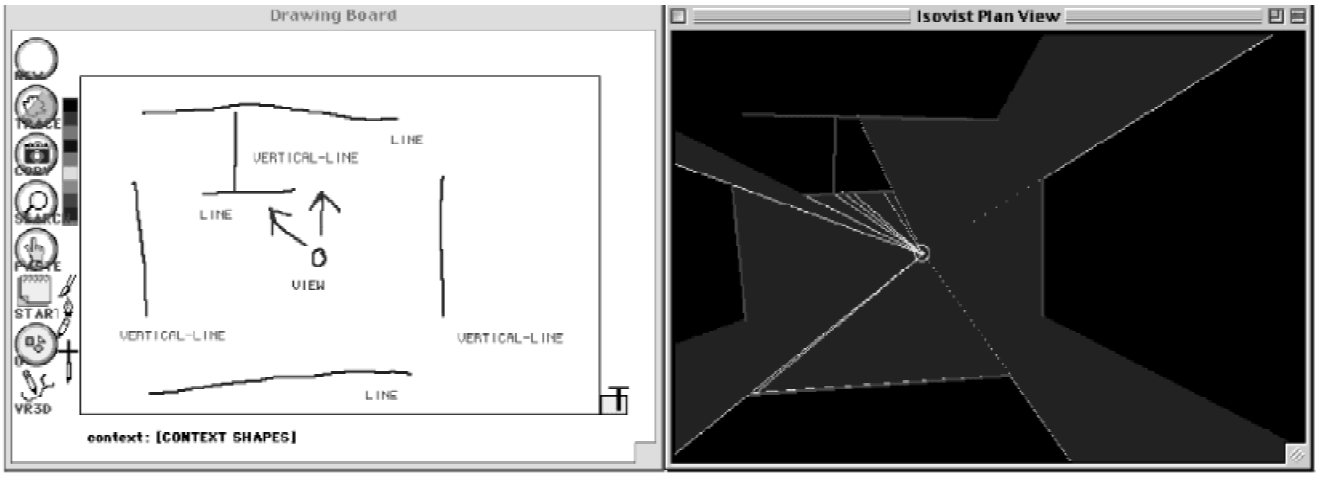


Fig. 30. The line drawings (left) are translated as wall lines and imported to the Isovist simulation program (right) to perform viewshed analysis (when a configuration of circles and arrows is recognized as a view symbol on the drawing board).

4.3.3. Visual analysis and simulation

Another example of a reactive drawing involves performing a visual analysis to test lines of sight on a floor plan. View shed analysis tools (Do & Gross, 1997) have been built around the idea called "Isovist" which defines the perception field of space through the calculation of visible area from a viewpoint (Benedikt, 1979, 1984; Davis & Benedikt, 1979). Figure 30 shows a module of the Right-Tool-Right-Time that would translate a hand sketched floor plan into an interactive simulation program that displays the evaluation and allows modification, which can in turn be brought back to the drawing environment for reference.

4.3.4. Testing and visualization through geometric representations

The active engagement of making drawing is also important in design reasoning. Freehand sketching is a freeform and direct interaction with drawing marks. For example, adding extrusion lines on a drawn floor plan engages the thinking and testing of form in 3-D space. Based on this observation, the Sketch-VR system (Do, 2001*b*) enables the designer to quickly sketch a floor plan and convert it to a 3-D model in Virtual Reality Modeling Language (VRML) or to draw symbol configurations to arrange furniture in the model.

As described earlier in Section 3, we found that designers draw 3-D representations alongside a 2-D floor plan to visualize and test how the space works. Designer would draw isometric projects of the floor plan sketch. Often they simply extrude the floor plan and make more than one 3-D sketch to see things in perspective (shown in Fig. 13). Figure 31 (left) shows a translation of a 2-D sketch into a 3-D geometric model so that the massing studies of objects can be viewed in different angles. Figure 31 (right) shows a

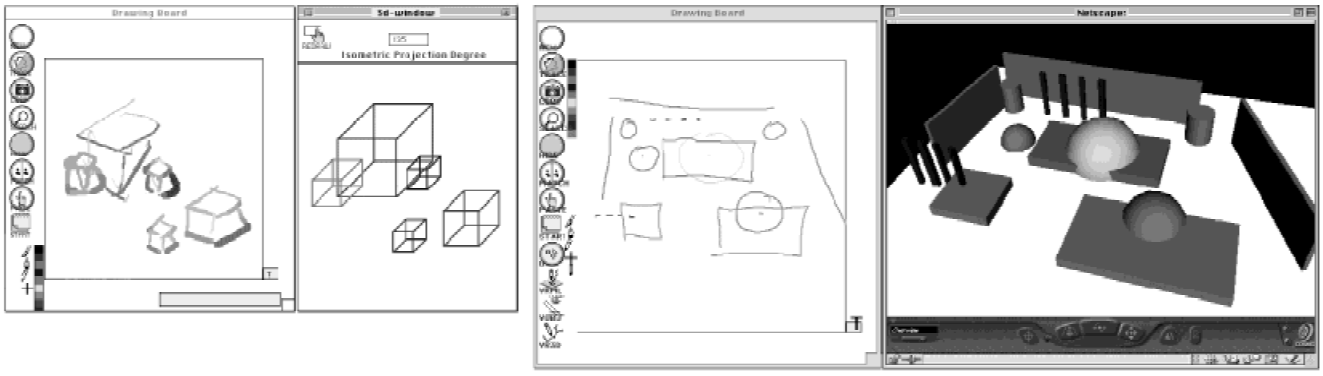


Fig. 31. From sketch to 3-D geometry. A drawing of massing studies is translated into a 3-D CAD model that allows changes of isometric views (left). A floor plan sketch is translated and displayed as a VRML model on a web browser (right).

https://doi.org/10.1017/S0890060402163037 Published online by Cambridge University Press



Fig. 32. Sketch-VR extrudes lines, circles, and curve shapes as wall lines to make floor plan partitions.

different approach to generating 3-D shapes, using Sketch-VR. In this case solid models such as boxes and spheres are placed in the VRML environment.

Figure 32 shows that in a simple scene, lines, curves and circles are extruded as partition walls and columns. Symbol configurations representing furniture elements are used to indicate placements for furniture layout. Arrows on the plan are recognized as gestures (instead of objects) to indicate views of interest and to set a guided path into the 3-D world (Fig. 33).

Finally, Sketch-VR provides functionality for generating a complex 3-D curved surface model by sketching the edge boundaries and the cross section of the surface. For example, Figure 34 (left) shows that by drawing three curves, Sketch-VR calculates and interpolates all the points on the surface and builds a mesh model. Making a complex curve like this in a CAD model requires comprehensive knowledge of various operations and manipulations of the control points of each curve. Recent architectural adventures, such as Gehry's Guggenheim Museum in Bilbao (Gehry & IDOM Associates, 1997) requires the use of complex and costly aerospace CAD software and 3-D digitizing hardware. Figure 34 (right) shows how such curved surface can be easily applied as a roof to provide shelter for a room or house.

5. DISCUSSION

5.1. Summary and future work

The study of reasoning in the design process has come to interest many design researchers and cognitive scientists. We believe drawing is the key element in the design process, and it should be carefully examined and used as the interface in computational tools to support design. By drawing, specifically we mean freehand sketching and diagramming. In contrast to the many computational tools that provide computational drafting capabilities, we believe it is important to keep the sketch form because it supports ambiguity and imprecision and invites reflection (reacting) and alternative readings of the drawing (marks).

This paper outlines the ideas, techniques and components that would enable the construction of a more human-oriented interface with which domain experts such as architects and designers can interact with various knowledge-based design systems. The computational sketching tools we present here are informed by the studies of design drawing we conducted. However, they did not attempt to only follow from the findings or to enlist features from them. They are intended to support designers in using sketching

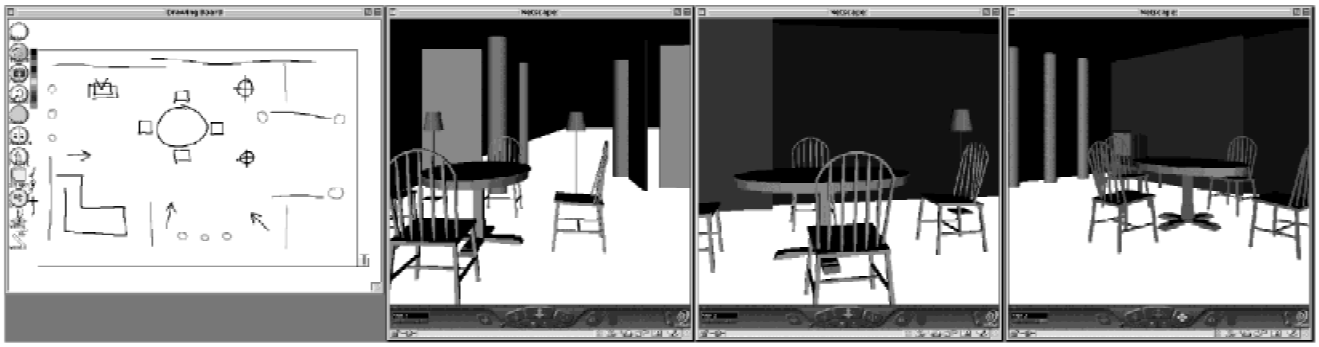


Fig. 33. A furniture layout sketch (TV, couch, table, columns, and walls) to create a 3-D VRML world. The arrows on the floor plan sketch are translated as viewpoints, which are embedded into the browser to provide a guided path into the 3-D scene.

https://doi.org/10.1017/S0890060402163037 Published online by Cambridge University Press

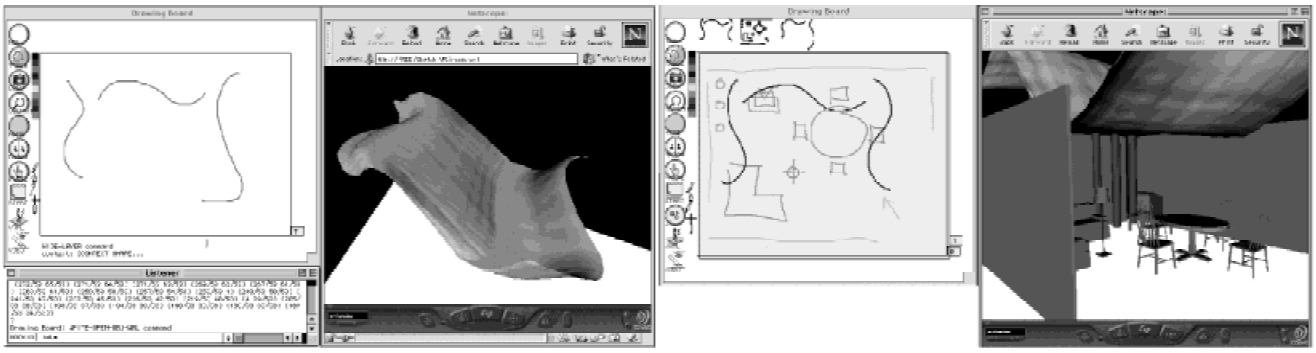


Fig. 34. A 3-D curved surface can be generated by sketching the two edge boundaries and a cross-section curve. For example, this type of curvy surface can be placed as a roof for a house or a shelter.

to interactively create, capture, inspect, index, and acquire design knowledge.

Many other features of drawing marks, acts, or reacts are worth exploring. Our computational tool also records speeds, pen pressures, and the drawing sequence as it recognizes drawing symbols. The sequence of drawing is a procedure to form or refine a design idea. This is a research area yet to be explored. If we argue that architectural design is ultimately about making decisions about 3-D space, then it would be valuable to draw directly in a 3-D scene. We are exploring sketching in 3-D scene (Jung et al., 1999; Jung & Do, 2000; Jung et al., 2001) to indicate concerns directly on the artifact in the space to facilitate collaboration.

In sum, sketching is important in the early, conceptual stages of architectural design, when designers are concerned with visual and spatial reasoning. Therefore, computational tools should support sketching activities. We described our empirical studies that examined the intentions in and relations among design drawings. The findings set up an interesting framework for research investigation of drawing: drawing marks, drawing acts, and drawing reacts. We believe designers draw to "see" and "move," and therefore it is important to study the marks, acts, and reacting drawing activities and to support them with computational tools. Our prototype systems demonstrate how freehand drawing interfaces can support various methods of analysis and design.

5.2. Representation and unstructured drawing for design and analysis

As described in previous sections, freehand drawing is an essential part of the design process. However, most systems that attempt to support drawing are actually structured drafting systems. Sutherland, although he built a drawing program that provides "clean drawing" admits that it is difficult to use Sketchpad to sketch because of its "structured nature" (Sutherland, 1975).

CAD systems supposedly support design. However, most current CAD systems are limited to drafting, modeling and rendering. Design professionals use CAD to create presentation and construction drawings or near-complete designs.

Those who claim to use CAD for design such as Giovannini and Gehry (Giovannini, 1993; Gehry & IDOM Associates, 1997) are actually using CAD to manipulate or represent the distortion and transformation of geometric shapes. Using CAD systems did not help these designers with their design reasoning. Most designers continue to develop solutions with paper and pen. They sketch to design.

What, then, should a computer program recognize from a drawing? Schodek suggests that an object in a computational representation should not just carry one class inheritance but should have multiple classes for different purposes (Schodek, 1994). For example, a column in a facade and a column in structural analysis would have different meanings, and should therefore have different representational foci. The interpretation of objects should change based on the different design purposes, yet most systems only offer one, standardized description for an object. Schodek points out that when applying AI techniques, to architectural problems we need to emphasize intent rather than mere problem solving. Schodek's comment suggests the need to detect design intention in a drawing.

If drawing conventions indicate intent and designers share drawing conventions, then a computer program should be able to recognize these intentions. However, it is also possible that some conventions and design knowledge could be so obvious or well known that they would not be represented in a drawing. This is when a knowledgeable computer tool can help supply the missing knowledge. Tools that perform visual field analyses, evaluate energy consumption, present past failures or success stories, help with scheduling and budgeting, or estimate material costs are important because they serve different purposes in design. In a design environment these "intelligent" tools and techniques could help as reminders or, consultants or could offer alternatives. If the designer's intentions are detected during drawing engagement based on the context, then the computer program could provide missing knowledge or the relevant

https://doi.org/10.1017/S0890060402163037 Published online by Cambridge University Press

tools. Designers can make better use of these tools if they are made available at the right time and with an appropriate interface.

5.3. Why should computer systems support sketching?

Sketching is a natural way for people to explain and understand complex ideas and to perform visual and spatial reasoning. Our goal is to develop computer systems that support sketching activities so that designers and knowledge-based systems can interact through sketching. In architectural design today, although many CAD programs support drafting, most conceptual and creative work is still done using the traditional media of paper and pencil. Pencil on paper is more flexible and easy to use compared with conventional CAD software. Using a pencil to draw allows the designer to explore more freely and quickly. With paper the designer can mark directly on the design drawing, indicating shape, line weight, color, and position without stopping to type commands or select menu items.

Designers draw what they want without the burden of learning obscure commands or being forced to specify a defined shape when they are only concerned about a rough object or space. Therefore, to support creative design work, CAD systems should emulate paper-based media and enable the designer to work with a pen in an unstructured way.

But what is the advantage to using computers to support sketching if the machine only mimics paper media? Some sophisticated painting programs (e.g., Painter, Photoshop, etc.) allow freehand pen input and display a result that is similar to a scanned image. To edit the painting, however, a user must work at the pixel level. Such programs are not intelligent with respect to what is drawn. To support creative design, a computational sketching environment might offer additional capabilities. An obvious need is more powerful editing, allowing the user to reshape lines, and to delete, group, and duplicate figures in sketch form as with conventional (structured, menu based) CAD drawing tools.

Beyond tools for making and editing sketches, computing environments for creative and early conceptual design should recognize sketch and diagram elements and provide simulations, critiques, constraint maintenance, and knowledge-based editing. More sophisticated enhancements could contribute even more useful functions to a computer-based sketching environment, giving it many advantages over paper. These environments could provide access to relevant information during the earlier stages of design, when changes in strategy are less costly. Designers might use the computer sooner in their design process and thus gain valuable feedback earlier in design.

ACKNOWLEDGMENTS

This research was supported in part by the National Science Foundation under Grants IIS-96-19856 and IIS-00-96138. The views contained in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

Akin, O. (1978). How do architects design. In *Artificial Intelligence and Pattern Recognition in Computer Aided Design, IFIP* (Latombe, E.J.-C., Ed.), pp. 65–104. New York: North-Holland.

Akin, O. (1986). *Psychology of Architectural Design*. London: Pion.

Baudel, T. (1994). A mark-based interaction paradigm for free-hand drawing. *ACM Symposium on User Interface Software and Technology*. Marina del Rey, CA: ACM Press.

Benedikt, M.L. (1979). To take hold of space: Isovist and isovist fields. *Environment and Planning B6*, 47–65.

Benedikt, M.L. (1984). Perceiving architectural space: From optic arrays to isovists. In *Persistence and Change* (Warren, H.W., & Shaw, R.E., Eds.), pp. 103–114. Hillsdale, NJ: Erlbaum.

Cross, N., Christiaans, H., & Dorst, K., Eds. (1996). *Analyzing Design Activity*. New York: Wiley.

Davis, L.S., & Benedikt, M.L. (1979). Computational models of space: Isovists and Isovist fields. *Computer Graphics and Image Processing 11*, 49–72.

Do, E.Y.-L. (1995). What's in a diagram that a computer should understand? *CAAD Futures '95: The Global Design Studio, Sixth International Conference on Computer Aided Architectural Design Futures* (Tan, M., & Teh, R., Eds.), pp. 469–482. Singapore: National University of Singapore.

Do, E.Y.-L. (1998). *The right tool at the right time—Investigation of free-hand drawing as an interface to knowledge based design tools*. PhD dissertation. Atlanta: Georgia Institute of Technology.

Do, E.Y.-L. (2001*a*). Graphics interpreter of design actions: The GIDA system of diagram sorting and analysis. In *CAAD Futures 2001* (Vries, B.d., van Leeuwen, J.P., & Achten, H.H., Eds.), pp. 271–284. Eindhoven, The Netherlands: Kluwer Academic Publishers.

Do, E.Y.-L. (2001*b*). VR Sketchpad: Creating instant 3D worlds by sketching on a transparency window. In *CAAD Futures 2001*. (Vries, B.d., van Leeuwen, J.P., & Achten, H.H., Eds.), pp. 161–172. Eindhoven, The Netherlands: Kluwer Academic Publishers.

Do, E.Y.-L., & Gross, M.D. (1995*a*). Drawing analogies—Supporting creative architectural design with visual references. In *Computational Model of Creative Design* (Gero, J., & Maher, M.-L., Eds.), pp. 37–58. Heron Island, Australia: University of Sydney.

Do, E.Y.-L., & Gross, M.D. (1995*b*). Shape based reminding as an aid to creative design. *The Global Design Studio, Proc. Sixth Int. Conf. on Computer Aided Architectural Design Futures* (Tan, M., & Teh, R., Eds.), pp. 79–90. National University of Singapore, Centre for Advanced Studies in Architecture, Singapore.

Do, E.Y.-L., & Gross, M.D. (1997). Tools for visual and spatial analysis of CAD models—Implementing computer tools as a means to thinking about architecture. In *CAAD Futures 97* (Junge, R., Ed.), pp. 189–202. Munich: Kluwer.

Do, E.Y.-L., Gross, M.D., Zimring, C., & Neiman, B. (2000). Intentions and relations among design drawings. *Design Studies 21*, 483–503.

Do, E.Y.-L., Or, S.-W.D., Carson, D.M., Chang, C.-S., & Hacker, W.C. (1994). *Usability Study of a Case-Based Design Aid: Archie*. Atlanta: Georgia Institute of Technology.

Domeshek, E., & Kolodner, J. (1992). A case-based design aid for architecture. In *Artificial Intelligence in Design '92* (Gero, J.S., Ed.). Dordrecht: Kluwer Academic Publishers.

Dorst, K., & Cross, N. (1995). Protocol analysis as a research technique for analysing design activity. *Design Engineering Technical Conf. 2(DE-83)*, pp. 563–570. New York: ASME.

Eastman, C.M. (1968). On the analysis of intuitive design. In *Emerging Methods in Environmental Design and Planning* (Moore, G.T., Ed.), pp. 21–37. Cambridge, MA: MIT Press.

Eisenman, P. (1987). *Peter Eisenman House of Cards*. New York: Oxford University Press.

Forbus, K.D., Ferguson, R.W., & Usher, J.M. (2001). Towards a computational model of sketching. In *Int. Conf. on Intelligent User Interfaces*, pp. 77–83. New York: ACM Press.

Fraser, I., & Henmi, R. (1994). *Envisioning Architecture—An Analysis of Drawing*. New York: Van Nostrand Reinhold.

https://doi.org/10.1017/S0890060402163037 Published online by Cambridge University Press

Galle, P., & Kovács, L.B. (1992). Introspective observations of sketch design. *Design Studies 13*, 229–272.
Gehry, F.O., & IDOM Associates. (1997). Project Diary: Guggenheim Museum Bilbao, Bilbao, Spain—An architect's dream project in Spain's Basque Country brings worldwide attention. *Architectural Record 185(10)*, 74–87.
Genau, A., & Kramer, A. (1995). Translucent history. *Human Factors in Computing Systems, CHI 95 Conf. Companion*, pp. 250–251.
Giovannini, J. (1993). Beyond convention. *Architecture: The AIA Journal 82*, 52.
Goel, V. (1995). *Sketches of Thought*. Cambridge, MA: MIT Press.
Goldschmidt, G. (1989). Problem representation versus domain of solution in architectural design teaching. *Journal of Architectural and Planning Research 6*, 204–215.
Goldschmidt, G. (1991). The dialectics of sketching. *Creativity Research Journal 4*, 123–143.
Graves, M. (1977). The necessity for drawing: Tangible speculation. *Architectural Design 6*, 384–394.
Gross, M.D. (1996). The Electronic Cocktail Napkin—Working with diagrams. *Design Studies 17*, 53–69.
Gross, M.D., & Do, E.Y.-L. (1996). Demonstrating the electronic cocktail napkin: A paper-like interface for early design. *CHI 96, Conference on Human Factors in Computing Systems, Conf. Companion*, pp. 5–6. Vancouver: ACM Press.
Gross, M.D., Do, E.Y.-L., & Johnson, B.R. (2001). The Design Amanuensis: An instrument for multi-modal design capture. In *CAAD Futures 2001* (Vries, B.d., van Leeuwen, J.P., & Achten, H.H., Eds.), pp. 1–13. Eindhoven, The Netherlands: Kluwer Academic Publishers.
Gross, M.D., Zimring, C., & Do, E.Y.-L. (1994). Using diagrams to access a case base of architectural designs. In *Artificial Intelligence in Design '94* (Gero, J., Ed.), pp. 129–144. Lausanne, Switzerland: Kluwer Academic Publishers.
Herbert, D.M. (1993). *Architectural Study Drawings*. New York: Van Nostrand Reinhold.
Jung, T., & Do, E.Y.-L. (2000). Immersive redliner: Collaborative design in cyber space. *ACADIA 2000, Associations of Computer Aided Design in Architecture National Conf*. (de Velasco, G.P.V., & Clayton, M., Eds.), pp. 185–194. Washington, DC: ACADIA.
Jung, T., Do, E.Y.-L., & Gross, M.D. (1999). Immersive redlining and annotation of 3D design models on the web. *Computers in Building (June, '99) Proc. CAAD Futures '99 Conf*. (Eastman, C., & Augenbroe, G., Eds.), pp. 81–98. Dordrecht: Kluwer.
Jung, T., M.D. Gross and E.Y.-L. Do (2001). Space pen: Annotation and sketching on 3D models on the Internet. In *CAAD Futures 2001* (Vries, B.d., van Leeuwen, J.P., & Achten, H.H., Eds.), pp. 257–270. Eindhoven, The Netherlands: Kluwer Academic Publishers.
Kahn, L.I. (1931). The value and aim in sketching. *T-Square Club 1*, 19.
Kolodner, J.L. (1991). Improving human decision-making through case-based decision aiding. *AI Magazine 12*, 52–68.
Kolodner, J.L. (1993). *Case-Based Reasoning*. San Mateo, CA: Morgan Kaufmann.
Kramer, A. (1994). Translucent patches—Dissolving windows. *ACM Symposium on User Interface Software and Technology*. Marina del Rey, CA: ACM Press.
Kramer, A. (1995). Supporting design activities in the written medium. *Human Factors in Computing Systems, CHI 95 Conf. Companion*, pp. 61–62.
Landay, J.A. (1996). SILK: Sketching interfaces like krazy. *Human Factors in Computing, CHI 96* [Television program].
Landay, J.A., & Myers, B.A. (1995). Interactive sketching for the early stages of interface design. *Human Factors in Computing Systems, CHI '95*, pp. 43–50. Denver, CO: ACM Press.
Laseau, P. (1986). *Graphic Problem Solving for Architects and Builders*. Boston: Van Nostrand Reinhold.
Lawson, B. (1994). *Design in Mind*. Oxford, UK: Butterworths.
Leclercq, P. (2001). Programming and assisted sketching. In *CAAD Futures 2001* (Vries, B.d., van Leeuwen, J.P., & Achten, H.H., Eds.), pp. 15–32. Eindhoven, The Netherlands: Kluwer Academic Publishers.
Matthews, K. (1994). *The Great Buildings Collection*. New York: Van Nostrand Reinhold.
McFadzean, J., Cross, N., & Johnson, J. (1999). Drawing and the soliloquising of design suppositions. *Design Thinking Research Symposium* (Porter, W., & Goldschimidt, G., Eds.), pp. 23–35. Cambridge, MA: MIT Press.
McKim, R.H. (1980). *Thinking Visually: A Strategy Manual for Problem Solving*. Belmont, CA: Lifetime Learning Publications.
Newell, A., & Simon, H.A. (1963). GPS: A program that simulates human thought. In *Computers and Thought* (Feigenbaum, E.A., & Feldman, J., Eds.), pp. 279–296. New York: McGraw Hill.
Porter, W. (1988). Notes on the inner logic of designing: Two thought-experiments. *Design Studies 9*, 169–180.
Robbins, E. (1994). *Why Architects Draw*. Cambridge, MA: MIT Press.
Saund, E., & Moran, T.P. (1994). A perceptually-supported sketch editor. *ACM Symposium on User Interface Software and Technology*. Marina del Rey, CA: ACM Press.
Schodek, D.L. (1994). Critical issues in automation based creative design education. *Automation Based Creative Design* (Tzonis, A., & White, I., Eds.), pp. 447–458. Amsterdam: Elsevier.
Schön, D.A. (1985). *The Design Studio*. London: RIBA Publications.
Schön, D.A. (1988). Designing: Rules, types and worlds. *Design Studies 9*, 181–190.
Schön, D.A., & Wiggins, G. (1992). Kinds of seeing and their functions in designing. *Design Studies 13*, 135–156.
Sivaloganathan, S. (1991). *Sketching Input for Computer Aided Engineering: Mechanical Engineering*. London: City University.
Stahovich, T.F. (1996). SketchIT: A sketch interpretation tool for conceptual mechanical design. Cambridge, MA: MIT Artificial Intelligence Laboratory.
Sutherland, I. (1963). *Sketchpad—A graphical man–machine interface*, Cambridge, MA: MIT.
Sutherland, I.E. (1975). Structure in drawings and the hidden-surface problem. In *Reflections on Computer Aids to Design and Architecture* (Negroponte, N., Ed.), pp. 73–85. New York: Petrocelli/Charter.
Suwa, M., & Tversky, B. (1996). What architects and students see in architectural design sketches: A protocol analysis. *First International Symposium on Descriptive Models of Design*, Istanbul, Turkey.
Taggart, J. (1975). Sketching: An informal dialogue between designer and computer. In *Reflections on Computer Aids to Design and Architecture* (Negroponte, N., Ed.), pp. 147–162. New York: Petrocelli Charter.
Ullman, D.G., Wood, S., & Craig, D. (1990). The importance of drawing in the mechanical design process. *Computer Graphics 14*, 263–274.
Van Sommers, P. (1984). *Drawing and Cognition: Descriptive and Experimental Studies of Graphic Production Processes*. Cambridge: Cambridge University Press.
Zeleznik, R.C., Herndon, K.P., & Hughes, J.F. (1996). Sketch: An interface for sketching 3D scenes. *SIGGRAPH '96*, pp. 163–170.
Zimring, C., Do, E.Y.-L., Domeshek, E., & Kolodner, J. (1995). Supporting case-study use in design education: A computational case-based design aid for architecture. In *Computing in Civil Engineering, A/E/C Systems '95* (Mohsen, J.P., Ed.). Vol. II, pp. 1635–1642. Atlanta, GA: American Society of Civil Engineers.

Ellen Yi-Luen Do is an Assistant Professor of Architecture at the University of Washington, Seattle, where she codirects the Design Machine Group. She received a Bachelor in architecture (Honors) from National Cheng-Kung University in Taiwan (1988), a Master in design studies from the Harvard Graduate School of Design (1991), and a PhD in design computing from Georgia Institute of Technology (1998) with a minor in cognitive science. She has conducted empirical studies of design drawing and constructed computer software to integrate knowledge-based applications with freehand drawing. She has also worked in the areas of computer-based visual analysis and case-based reasoning tools.

https://doi.org/10.1017/S0890060402163037 Published online by Cambridge University Press