Graphics Interpreter of Design Actions
the GIDA system of diagram sorting and analysis

Ellen Yi-Luen Do
Design Machine Group, Department of Architecture, University of Washington, Seattle, WA 98195-5720, USA

Key words: diagrams, design action, graphics interpreter, design thinking, diagram sorting

Abstract: This paper describes the background and research for a diagram sorting and analysis system called GIDA – Graphics Interpreter of Design Actions. The GIDA system explores how spatial relations between graphic symbols in a design drawing may be extracted and the transformations of the graphic elements between different drawings in a design process may be tracked.

1. SCENARIO

Imagine one day you wake up to find that you do not remember who you are and what you do. A recent accident has injured your brain but left your motor skills and perception skills intact. You learn that you used to be an architect and designed many houses. Of most significance, you had been involved with a mental exploration, a house design for a single residence for the past ten years. All the design drawings are archived in paper format as well as scanned images stored on half a dozen CD-ROMs.

To help yourself get back your edge and to relearn how to be a designer, you decided to examine closely all these drawings to discover the relationships between the drawings, and to learn how one designer works, in this case, how you used to design. Your personal notes showed that the design was inspired by Le Corbusier’s ‘five points of architecture’ principle and the idea of ‘place within a place’. Everything seems easy, you have abundant drawings to examine, except for one problem: these drawings have
no date or time information and therefore no sequential reference can be made.

Now, look at Figure 1. Here are the drawings you pulled out from the archive and arranged them side by side on the table. You told yourself, this is easy: they are all projections of the same building: isometric, plan, section and elevation (assuming you remember what those different drawing types mean). Look again. Obviously they belong to the same family of the design drawing for the residential house because they share several design elements (Thick walls, hood, bridge, balcony, columns, chimney box and pipe). However, upon a closer look, there seem to be variations among the drawings. Some design elements were moved, rotated or height and width adjusted. This is fun, you think. If you can learn the transformations between design elements, and procedure of design moves, then you would learn how to design by following certain procedures.

Figure 1. Various drawings of a pavilion house design

This is no fairy tale, nor do we wish anyone to be a victim of this nightmare. We are interested in developing computational tools to help us learn how architects design: specifically, the ways or arrangements, and transformations of design elements. Our previous work (Do, Gross et al. 2000) investigated the transformations of design elements in a series of drawings. We looked at the boundaries of the graphics space of these design actions, extracted and defined the rules that we could identify in the actions and products that it encompasses. The previous work suggested that a computational system might be built to emulate and analyse these actions.

In the following sections, we first present the background of this investigation of procedural knowledge in design process. Then section three examines related work on design cognition, including our previous work on spatial reasoning and a coding scheme. Section four describes the implementation of a graphics sorting and analysis system called GIDA explaining system architecture and functionality. Finally, we discuss some issues of analysing design drawings, and future research directions.
2. BACKGROUND

Recently we have seen a growing interest among cognitive scientists and design studies researchers in studying design drawings and diagrammatic reasoning. Researchers use methods such as think-aloud protocols (Akin 1978; Akin 1986; Akin and Lin 1995; Cross, Christiaans et al. 1996), retrospective analysis of design behaviour (Suwa and Tversky 1996), introspection (Galle and Kovács 1992) and analysis of design products (Schön 1983; Porter 1988) to study the role of drawing in design.

Drawing provides representations for problem solving, idea generation and exploration of design alternatives. Surprisingly, although drawing is the subject of this research studies have mostly focused on the verbal transcripts instead of the graphical representations that designers produce. Instead, we focus on design drawings themselves and propose a computational system to help identify and analyse connections between different design drawings.

In this paper, we report on the functionality and the implementation of a computer based graphical interpreter. We describe how this diagram sorting and analysis system can help design researchers to explore the space of graphic actions in which design intentions are expressed. We conclude with a discussion of future research directions.

3. RELATED WORK

3.1 Drawing and Design Thinking

One popular model of design studies draws a parallel of design with problem solving and views design as information processing (Newell and Simon 1963; Newell and Simon 1972). Design studies researchers and cognitive scientists use protocol analysis (Eastman 1968, Akin, 1978 #1580; Akin 1986) to study design as a process of problem formulation and solution generation. Transformations (links) of different states (nodes) are illustrated using Problem Behaviour Graph (PBG) notation. Chan’s use of schemas (Rumelhart and Ortony 1977; Rumelhart 1980) represents domain-specific knowledge as design constraints and associated rules in memory (Chan 1990). Moran proposed that design (Moran 1970) consists of memory, representation conventions, interpreted problems, and design strategies. He argues that designer use many kinds of representations, or “languages” to express the state of the problem. The information processing model of
design is also extended by researchers (Oxman and Oxman 1992) to include model-based refinements and case-based adaptations.

Design researchers also analyse the modalities of design actions. Schön argued that designers develop rules to guide their own thinking process (Schön 1988). He defined design as an act of ‘reflection-in-action’ (Schön 1983). Designers ‘see’ and then ‘move’ (Schön and Wiggins 1992) their design objects—through the act of drawing. Goldschmidt further suggests drawing is ‘interactive imagery’ (Goldschmidt 1991) and that designers use design drawing to perform ‘seeing as’ and ‘seeing that’ reasoning modalities (Goldschmidt 1989; Goldschmidt 1991). She argues that design, or design image transformation, is a systematic dialectic with oscillation of arguments carried out through the act of sketching.

Besides ‘think-aloud’ protocols, another type of studies uses introspective, retrospective or speculative knowledge. Galle and Kovács (Galle and Kovács 1992) argue that an introspective record provides ample time for reflection and eliminates the need to rely on either an ‘information processing model’ or other type of assumption for analysis. They argue that introspection is a useful supplement to either protocol or interview studies that often happen in a short period of time. Suwa and Tversky applied retrospective reports of design sessions (Suwa and Tversky 1996) to study designers’ perceptual processes in observing their own sketches. They cited Dorst and Cross’s review paper (Dorst and Cross 1995) to argue that think-aloud protocols has the disadvantage that concurrent verbalisation and behaviour could cause side effects or account for incomplete activities.

Porter conducts a “thought-experiment” using speculative re-construction of design process (Porter 1988) to account for the underlying logic of designing. He presents a chain of reasoning—arguments about how the design might have evolved—for two cases, an existing plaza and a building design from conception to their present state. Porter argues that a skilled designer or observer will discover design characteristics from its environmental settings—the appreciation of a site and his own reasoning of design principles. Therefore, a ‘replication’ of design process is a form of inquiry for design teaching and for revealing the implications of computer tools.

Architectural historians share this notion of relationships between design and its drawing. Hewitt argues that the history of architectural drawing is “a medium of thought” (Hewitt 1985). He argues that an ‘idea sketch’ consists of “personal and intuitive, or may be based on clearly defined methodologies or programs of instruction.” This ‘conception’ of design is “a triad of interrelated operations—thinking, seeing, and drawing.” Interestingly, a recent study by Akin and Lin echo this argument (Akin and Lin 1995). Their design experiment involved subjects reproducing a drawing from a printed
transcript, and predicting the verbal data from a video of the design drawing process with the audio track suppressed. They conclude that novel design decisions usually occurred when the designer was in a “triple mode period” of drawing, thinking and examining.

3.2 Related work on graphic systems

Many design studies research and computer aided design projects have attempted to provide better support for classifying graphics representation, automatic generation of graphic shapes, and recognition of building elements. For example, Electronic Cocktail Napkin (Gross 1994; Gross 1996; Gross and Do 1996) takes freehand sketching input from a digitising tablet and stylus and recognises shapes and diagram configurations. Each drawing mark has time stamp, pressure and speed information. Users of the system can define different drawing symbols with single or multiple strokes, as well as combinations of symbols to form diagrams. The Napkin project also has a replay function that can display drawing marks stroke-by-stroke in an animated sequence. The Design Amanuensis (Gross, Do et al. 2001) is a system that records voice input in parallel with recording drawing actions. Any segment of the design session (i.e., audio or the automatically recognised transcripts) can be later played back along with the drawing sequence. Computational Sketch Analysis (CSA) system (McFadzean, Cross et al. 1999) uses video cameras to record designers' drawing and physical activities. The graphical notational activities can be later replayed and used to build structures of the notations according to a predefined schema. HyperSketch I & II (McCall, Johnson et al. 1997; Gross, Do et al. 1998) record design drawings using a digitising tablet and generates a relationship coding (e.g., ‘trace from’ and ‘design alternative’) of designers' freehand drawings. It represents and links the drawings as nodes in a hyper document graph.

In addition to the research projects on recording design drawings, there are interesting efforts to identify graphic units in design drawing and automated recognition and reasoning of shapes. For example, Wang in his “Ways of Arrangement” presented a coding scheme (Wang 1987) focusing on the spatial relationships between objects (e.g., abut, adjacent). Our scheme (Do, Gross et al. 2000) on the other hand, focuses on the transformations of objects (design elements or drawings) among different states (e.g., staircase moved from east to west, wall height reduced). Achten’s Generic Representations (Achten 1997) surveys architectural drawings to analyse the relations between graphic marks and design decisions. He classifies all design representations into one or a combination
of three basic themes: ‘shape’, ‘structure’ and ‘system’. For example, ‘simple contour’ and ‘specified form’ concerns the theme of ‘shape’. Likewise, the types of generic representation such as ‘modular field’, ‘schematic axial system’ and ‘proportion system’ all dealing with the theme of ‘the structure underlying shapes’.

Goel’s examination of designers in action reveals that the structure of drawing helps facilitate cognitive transformations (Goel 1995; Goel 1999). He argues that design process corresponds to ill- and well-structured representations and therefore non-notational and notational representations are both used. Specifically, the study of an experienced architect with right brain injury shows that drawing skills and the ability to design are ruled by different parts of the brain. In this example, even though the patient’s drawing skills and explicit architectural knowledge base was intact, he could not make as many vertical and lateral transformations as a an uninjured architect. (Lateral transformation means the ability to generate design options between plan and sections, and variations of design of the same projections, i.e., a plan or sectional view).

Another direction of research is automatic recognition and reasoning of design drawings. For example, a computer vision technique was used to automate plan recognition (Koutamanis and Mitossi 1993). Design floor plans are defined as a configuration of architecture elements (i.e. walls, windows, and doors) and were used as templates for optical character recognition (OCR). With automated recognition of building elements and spatial subdivision, further analysis such as topological description can then be deducted. Gero, et al. develop a series of qualitative representation encoding scheme to attempt reasoning about two-dimensional and three-dimensional shapes (Gero and Park 1997; Gero and Damski 1999; Gero 1999). Their approach treats simple geometric shapes as exemplars of classes of shapes that can be represented by the vertex angle (A), length of edge (L) and curvature (K) which they call Q-codes. Using this coding scheme, drawing symbol configurations can then reveal the qualitative features such as repetition pattern or symmetry.

Our GIDA system is well situated in this family of design study research. The coding scheme that follows was developed from analysing an architect’s drawing for a pavilion house design. The GIDA system uses the Electronic Cocktail Napkin as the graphic parsing and recognition engine. It extends the Napkin program with capabilities of comparing the state transformation between drawing symbol configurations, not just between drawing elements.
3.3 Our previous studies of design drawing

Any two drawings in a design project may share various properties: They may employ the same projection (plan, section, isometric), the same medium (crayon, pencil and pen), or exhibit the same design intentions (circulation or views). They may describe the same elements (bridge, columns, and strip windows) in different configurations. They may be constructed from the same view angle.

In a previous study (Do, Gross et al. 2000) we looked at an architect's collection of over a hundred drawings for a residential house design and developed a coding scheme to classify these drawings. The scheme codes properties of the drawings such as the elements depicted and projection type and view angles. The notation system focuses on state transformations of design elements from one drawing to another (e.g., stair moved from east to west, wall height reduced) as well as changes of view and projection type. The codes facilitate easier comparison and sorting of element types and operations. However, the amount of descriptive data—the number of types and fields associated with each drawing quickly becomes difficult to manage. Furthermore, it is hard to keep track of the sorted design elements and their source drawings. Therefore, we built the GIDA system to help manage and sort design drawings by their features and transformations.

4. GRAPHICAL INTERPRETATIONS

GIDA (Graphical Interpreter of Design Actions) is a graphic spreadsheet sorting program for researchers and designers to identify and analyse individual drawings and their symbol system, as well as the transformations of composite elements in a series of drawings. It was built to facilitate easy sorting of graphical design actions among different design drawings. Designers can use the GIDA system to encode design drawings (such as the notation system described above) by making diagrams of different graphic shapes (circle, line, triangle, etc.) and entity features (i.e., colour, orientation, etc.) by tracing a design drawing. A researcher can assign identifiers to each individual element drawn.

Currently, the GIDA interface only allows two-dimensional input. However, the three-dimensional internal representation of a building can be formed by linking different projections (e.g. top, front, and side views) together. Figure 2 below shows that a three-dimensional model of the building (with design elements) is constructed and serves as a base template for recognising and comparing graphic elements.
Figure 2. Two-dimensional projection representation of the three-dimensional building onto different planes (top, front, side).

The graphics engine of the program provides facilities to recognise simple shapes (circles, boxes, spiral) and configurations (overlapping boxes, crosses and T-junctions). The GIDA system then identifies the relative positions of elements in a drawing (on a 3x3 grid) and provides diagram-matching functions based on element types, element counts, and spatial relations. The transformations of elements among different drawings can also be identified and used as sorting criteria.

Figure 3. Picture underlay with analytical diagram and location identifier on a 3x3 grid

At the most basic level, GIDA can load any graphic file as an underlay base for diagram coding. After tracing and extracting the important design
elements, designer can call up GIDA’s local position identifier (LPI) to analyse the locations of each graphic shape. The LPI adjusts the area for analysis to the smallest bounding box that contains all the drawing and then apply a 3x3 grid over the area. Then it generates the list of the cells through which each graphic element travels. Figure 3 shows an underlay of a church floor plan, the designer’s analytical spatial diagram, and the top level LPI grid. For example, the top left circle of the church entrance corridor occupies cell 1 and 4, while the big circle representing the inner altar space occupies grid cells 2, 3, 5, 6, 8 and 9.

GIDA can also perform automatic feature extraction, or filtering. Figure 4 shows how a floor plan sketch (left) is extracted as a simple diagram configuration after filtering out overlapping lines and small drawing marks (right). The LPI can analyse both complex and simplified drawings and generate location lists for comparison. The goal is to identify the actions that transform a design, for example, object A was moved from location B to C. Therefore, extracting the diagram from the detailed design would significantly reduce the computation time and the generation of many uninteresting, low level object transformations.

![Figure 4. The filter function extracts sketch lines (left) to configuration of simple shapes (right).](image)

Besides analysing the locations of graphic elements and the state change between any two drawings, GIDA has an analogy transfer function to apply a transformation of spatial relations from a source pair (A B) to a new drawing (C D). Figure 5 shows that given the example of a-window (big box containing a small circle) transformed to b-window (small circle moved from concentric to right-of the box), for any given c-window (big circle containing a small box), the system would generate a new resulting d-window (small box moved from concentric to right-of the big circle) through
this process of analogy transfer. This analogy transfer function would be useful to infer or predict possible design moves based on the design transformation operations found in the drawing analysis.

![Figure 5. Analogy transfer, the transformation of spatial relationships from a-window to b-window is automatically applied to c-window to generate d-window.](image)

Figure 5. Analogy transfer, the transformation of spatial relationships from a-window to b-window is automatically applied to c-window to generate d-window.

Figure 6 left shows a pair of design drawings with frontal isometric views. Concerned about how element configurations are transformed between the two frontal façades, we could bring them in as underlay pictures and draw analytical diagrams on top of them. Figure 6 right shows the elevation diagrams traced over the pictures with underlay removed and LPI grid overlay. The GIDA system then generates a list of occupied cell numbers (drawn sequence) for each object, as shown in Table 1.

![Figure 6. Design drawings of isometric views (left) and their diagrammatic analysis (right).](image)

Figure 6. Design drawings of isometric views (left) and their diagrammatic analysis (right).

In this example, one drawing has nine elements and the other one has eight. Each element has a position in the global coordinate system and a list of the LPI 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 be (7 4 1 2 3 6 9 8 7). The sizes (bounding box) of this element in the two different
drawings are very close. Therefore, GIDA system inferred that the transformation for this element from the first drawing to the second one is a shifting up (addition of grid cells 1 2 3). The Hood element in drawing one is removed from drawing two. Likewise, the transformation of the Chimney Box is a moving down from (4 5 4) to (7 4 5 8 7).

Table 1. State of drawing elements represented as list of cell sequence.

<table>
<thead>
<tr>
<th>Element</th>
<th>Drawing #1</th>
<th>Drawing #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thick Wall</td>
<td>(7 4 5 6 9 8 7)</td>
<td>(7 4 1 2 3 6 9 8 7)</td>
</tr>
<tr>
<td>Chimney Box</td>
<td>(4 5 4)</td>
<td>(7 4 5 8 7)</td>
</tr>
<tr>
<td>Chimney Pipe</td>
<td>(1 4)</td>
<td>(1 4)</td>
</tr>
<tr>
<td>V-window 1</td>
<td>(6 9)</td>
<td>(3 6 9)</td>
</tr>
<tr>
<td>V-window 2</td>
<td>(5 1)</td>
<td></td>
</tr>
<tr>
<td>V-window 3</td>
<td>(4 7)</td>
<td></td>
</tr>
<tr>
<td>Hood</td>
<td>(6 3 2)</td>
<td></td>
</tr>
<tr>
<td>H-strip 1</td>
<td>(4 5 6)</td>
<td>(4 5 6)</td>
</tr>
<tr>
<td>H-strip 2</td>
<td>(4 5 6)</td>
<td>(4 5 6)</td>
</tr>
<tr>
<td>H-strip 3</td>
<td>(4 5 6)</td>
<td>(4 5 6)</td>
</tr>
<tr>
<td>H-strip 4</td>
<td></td>
<td>(7 8 9)</td>
</tr>
</tbody>
</table>

5. DISCUSSION AND FUTURE WORK

The GIDA system is a prototype, and a start for a larger research agenda on drawing and design action analysis. Many issues are worth exploring. First, although the current GIDA system can generate relations among any two graphics shapes or the transformation of a shape between any two drawings, the lists of relationship are too many and too detailed to be really useful. The obvious next step is to enable users to select a shape or a configuration and assign them specific naming classifications (e.g., parti, spatial themes or room names). Second, it's important to develop an easy to use interactive interface for selecting objects in different drawings for comparison. For example, an interface for designer to select the object in question in two drawings by picking or highlighting could be useful.

With this initial success of graphic action interpretations, we are musing over several interesting questions. Currently, we deal with transformations in only two-dimensional views. This is not surprising, considering that architects have been using orthogonal projections as a conventional way to communicate with others and to represent planning, for contracting construction, and to quickly generate alternatives. However, our building environment is three-dimensional and designer’s design operations are also three-dimensional. What would be the coding scheme if we extended the three-by-three grid over to three-by-three-by-three, for a total of twenty-

seven cell locations? Or shall we change the 3x3 to be 2x2 so that we can always subdivide a cell in half to get a finer detail analysis and comparison? It would be interesting to apply the recursive decomposition to drawing. We will look into the literature and research on quadtrees and octrees in the field of computer graphics and image processing. Another question is: should the GIDA system be equipped with automatic three-dimensional geometric object generation abilities so that design transformations can be easily visualised? Digital Clay (Schweikardt and Gross 1998) and VR Sketchpad (Do 2001) are some systems that could be a useful extension for GIDA.

Figure 7. Variations of three-dimensional coding scheme using grid cells: slices of side and top views of 3x3, total consideration of a 3x3x3 cube grid system, and quadtrees with infinite subdivision by half to get finer comparisons.

In conclusion, the GIDA system is a proof-of-concept application and a vehicle for design researchers to investigate the graphical operations that designers use. We built the GIDA system to help identifying re-occurring pattern of design manipulations in a design process. Further functionality can be added to the system so that it could record and analyze design actions in a protocol analysis. The GIDA system is useful for analyzing and comparing design actions between drawings of the same project, different projects, or any drawings that can be illustrated as simple diagrammatic representations. It could also be a tool for students to learn about design by studying the sketches of others. Or it could be a tool to help designers reflect during the design process.

6. ACKNOWLEDGEMENTS

This research was supported in part by the National Science Foundation under Grant numbers IIS-96-19856 and IIS-0096138. The views contained in this material are those of the author and do not necessarily reflect the
views of the National Science Foundation. The author would like to thank Bennett Neiman for providing his pavilion house design drawings as experiment example for the project.

7. REFERENCES

Goldschmidt, G., 1989, Architectural sketching, seeing as and seeing that, Unpublished manuscript submitted to the National Science Foundation.


