DESIGN SIMULATION

CHIU-SHUI CHAN and TODD R. BROWNING*
Department of Architecture, Iowa State University
482 College of Design, Ames, Iowa 50011, USA
**RTKL Associates Inc.*
140 South Dearborn St., Suite 200
Chicago, Illinois 60603, USA

Abstract. This paper intends to explore methods of constructing a design simulator. Two methodologies, approached differently, imitate the human design processes. The first component is an algorithmic method which has a cognitive model embedded. This cognitive model hypothesizes that human design has certain design logic applied. The design rationales are based on knowledge stored in a designer担 memory. Each time a similar design task is encountered, the same design procedures will be repeated for completion. What makes the results different are the design information used and sequences of processing it. A kitchen design using procedural algorithms is developed to simulate this design aspect. The second component simulates an intuitive design approach. Intuition is defined as design by rules of thumb, or heuristic design. This study investigated how to simulate an intuitive design process. The method involves building up a set of inductive rules symbolizing cultural aspects that need to be addressed in a design. A residential foyer design is the simulation task. The driving force is the heuristics. Results in this study have shown that there are many variables to include but impossible to capture and simulate any of the design processes, which are the reasons why studies in this area are difficult.

1. Introduction

In the area of design studies, many efforts have contributed to the development of design simulation by exploring possible algorithms to run design on computers. This research trend has been guided progressively by concepts on artificial intelligence (Eastman, 1973), shape grammar (Stiny and March, 1981), and case-based reasoning (Kolodner, 1993). Faced with the increasing influence of high-tech development in information technology, future design will encounter significant changes caused by the use of computers. Potential changes could result from any or all of the following: (1) the use of computer-aided design (CAD) applications for presentation; (2) the use of the Internet for communication between a design office and the construction site; (3) the use of the Web for promoting design products and conducting the bidding process; (4) the customization of CAD systems to develop a special tool for a firm to meet specific challenges for a project. Therefore, it is more important to explore how to put design into computers than to put computers into design.

To put computers into design is to manipulate CAD software while doing design. To put design into computers involves implementing design by a CAD system to finish the design processes. Specifically, it is to automatically execute or simulate some standard design by computers. There are two categories of design simulation. The first category is to simulate certain domain-specific design knowledge -- providing design assistance, for instance, to evaluate building performance, predict energy conservation, and analyze structural components. This type of simulation is not very difficult, because the engineering aspects of structure, energy conservation, and building conditions have certain well-defined procedures that can be represented by algorithms and mathematic operations. As long as the procedures are identified by mathematic formula, it is possible to complete the simulation.

The second category is to simulate the way designers think and do design, which usually applies techniques from artificial intelligence and cognitive science to establish design expert systems. This category involves complex cognitive patterns which make it difficult to (1) find appropriate representation for design knowledge to be executed in computers; (2) identify adequate structures to manipulate design rules; and (3) set up efficient processes for executing the operations. Therefore, most research in this area is slow and unproductive. In order to break this barrier, this research serves as a pilot study to experimentally explore methods for generating an architectural design by a CAD system.

Two themes symbolizing opposite design approaches have been the focus of this research. The first theme is to treat design as a rational process of solving problems. The problem solving paradigm began in the 1960s and early 1970s. Even though this approach has been shown to have many shortcomings in design (Archea, 1987), it still involves applying logic to solve problems in design activities; therefore, doing a design could be seen as solving a design problem. The complexity of a problem determines the sensitivity of skills needed for solving the problem. As a starting point, this study focus on simulating a low level of design expertise, which is to merely accomplish functionality by assigning some logic and fixed rules for decision making.

The second approach is to see design as an intuitive activity. In the real world, professional designers face problems that do not have easily formalized or algorithmic solutions. Heuristic methods must then be used. This requires searching for effective solutions depending on the timely use of knowledge to identify potential decisions that are promising and to rule out unpromising ones. Thus, the second theme is to explore the feasibility of utilizing a number of heuristic methods to test the possibility of simulating intuition. These two examples provide stimulation and inspiration for design automation and design tool development.

2. Example one: Design as solving a problem

The first simulation program is a procedural approach oriented to problem solving. The main structure is algorithmically driven and lineal step wide progress. The basic concept is based on certain design rules of functionality to generate a two-dimensional design. The program was written in AutoLISP and results were displayed in the AutoCAD environment.

Two key components for program construction are applied here to reflect a similarity to human design. The first component is a knowledge base representing design expertise. Conceptually, the more qualitative and quantitative professional information stored in the knowledge base, the more knowledgeable the designer is. The second component is a set of processing methods signifying design methodologies applied by designers. The premise is, the more efficient methods utilized at the right time at the right stage in the process, the more skillful the designer is. Of course, there are various ways of approaching design (Schon, 1983). This first study intends to test how large a knowledge base and how efficient a processing method are necessary to generate an acceptable design solution. As a pilot study, this program works on a simple kitchen design with a limited set of appliances, a simple set of functionality requirements, and no surrounding site conditions in order to simplify the task.

2.1. THE COMPONENT OF KNOWLEDGE BASE

Theoretically speaking, human design knowledge involves four categories of information stored in memory: analogy, fact, belief, and heuristic. Analogies are previous design solutions generated by the designer or learned from examples and stored in memory for future use. Facts are functional conventions as existing truth or functional requirements that relate to declarative knowledge which are “knowing that.” For example, the functional correlation between building elements for making spatial arrangements are functional facts. Beliefs are social, cultural, and environmental norms that are applied and addressed at any design stage. Heuristics can be thought of as "rules of thumb" – problem-solving strategies of means-end analysis, forming subgoals, and working backward (Newell, Shaw, & Simon, 1962; Newell & Simon, 1972). Heuristics often tend to lead to solutions but do not guarantee success. Contrasted with heuristics are "algorithms," which are methods and procedures that do guarantee a solution if the problem solver follows the steps correctly. Heuristics and algorithms both can be classified as parts of procedural knowledge which are "knowing how." In design, heuristics are regarded as any design principle or device that would reduce the average search for a solution.

All these forms of information create a huge knowledge base which varies

from person to person. The more abundant the information is, the higher the level of expertise the designer has. Of course, the quantity of data stored in human memory does not ensure its output quality. In fact, the quality of the information and the sequences of processing information have a great effect on output quality. In this study, only the functional facts and a limited number of heuristics are simulated.

2.1.1. Functional facts

The functional facts in this study are a set of default dimensions of all kitchen appliances and the functionality of spatial relationships among appliances. There are five essential units of range, sink, dishwasher, refrigerator, and counter area that define a residential kitchen. The standard dimensions of each unit are stored in a global variable as shown in Figure 1. For instance, the first item on the list is the range, with the following dimensions: 2'-6" by 2'-2". Of course, any of the appliances' dimensions may vary due to manufacturing styles.

```
(setq dimension '((range ((2 6) (2 2)))
                  (counter ((3 0) (2 2)))
                  (sink ((2 9) (1 10)))
                  (dishwasher ((2 0) (2 2)))
                  (refrigerator ((2 9) (2 4)))))
```

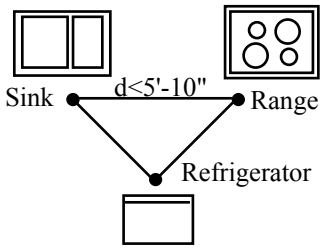


Figure 1. The dimensions of appliances and the conventional kitchen triangle.

The second set of information about functional facts is the spatial relationships among units. All spatial relations are encoded on a LISP list data format to distinguish whether a design unit should either be adjacent-to or across from the next unit. A kitchen design convention known as the kitchen triangle is used to determine efficient layout. This convention is also a heuristic that uses the correlated sequences of kitchen activities to determine the adequate distance which users have to walk between sink, range, and refrigerator. These functional conditions are critical to modern domestic kitchen planning when there is no counter island located in the middle of the room. Metaphorically speaking, lines joining these three elements of sink, range, and refrigerator form the known "work triangle" (see Figure 1). According to convention (Neufert, 1970, p. 54-56), the distance between sink and range should not exceed 1.8 meters (5'-10") and the total of the three triangle sides should be between 5.5 meters (18'-4") and 6 meters (19'-8"). Utilizing these numbers as basic criteria, the minimal size of the kitchen space can be determined. The kitchen triangle is defined by the relation across the three units of sink, range, and refrigerator as shown on the list

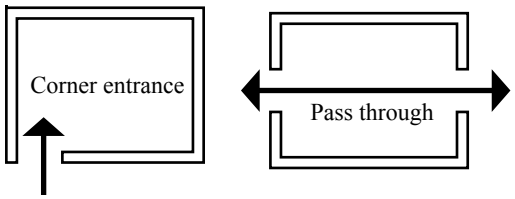
in Figure 2. This set of information is stored as a global variable, and is the mixed representation of facts and heuristics.

```
(setq relation
      '((range ((across counter) (adjacent-to counter) (across sink)
                (across dishwasher) (across refrigerator) (adjacent-to refrigerator)))
        (counter ((across range) (adjacent-to range) (adjacent-to sink)
                (across dishwasher) (adjacent-to dishwasher) (across refrigerator)))
        (sink ((across range) (adjacent-to counter) (adjacent-to dishwasher)
                (across refrigerator)))
        (dishwasher ((across range) (adjacent-to counter) (across counter)
                (adjacent-to sink) (across refrigerator) (adjacent-to refrigerator)))
        (refrigerator ((across range) (adjacent-to range) (across counter)
                (across dishwasher) (adjacent-to dishwasher) (across sink)))))
```

Figure 2. Representation of functional facts.

2.1.2. Design constraints

A design constraint is defined as certain requirements that must be fulfilled in order to solve a design problem. Such requirements could be any cultural, socio-political, organizational, financial, technical, mechanical, structural, legal, experiential and perceptual aspects of needs for a design unit or a group of units. In design, any constraint can be used to limit and reduce the search for possible solutions within the immense problem space. In this program, circulation is the constraint for determining the locations of openings that affect the internal working circulation and the accessibility to the kitchen area. The interior working circulation is defined as a working corridor with 6' depth between two rows of the units. The accessibility to the kitchen has two constraints. Option one locates the entrance door on the lower left corner, and the other option is a horizontal opening from left to the right as shown in Figure 3.




```
(setq designgoal '((range 1)
                   (counter 1)
                   (sink 1)
                   (dishwasher 2)
                   (refrigerator 3)))
```

Figure 3. The basic shape of the kitchen space and the predefined goal list.

2.2. GENERAL PROCESS AND USER INPUT FOR DESIGN DECISION MAKING

A design process is a sequential progression from the beginning of a design to the stage at which a product is generated. The entire process can be broken down into a sequence of goals symbolizing a number of stages of accomplishing one

or several tasks. Thus, the progress of this simulation is represented by a waiting list to execute the design units. This waiting list is presumed to be a priority list determined by the user. Theoretically, this list reflects the importance and preference of the design units to the designer. It also signifies a logical progression, instead of random attempts. The sequence of priority is by user input. For instance, if the user's input is refrigerator, sink, range, counter, dishwasher, then the waiting list is formed in this order.

After the priority list is defined, the access door constraint is requested to determine the location of the openings. The program will check the priority list to see whether there would be a potential conflict for arranging the units. The possibilities are the overlap or gap between the placement of units. If any conflict exists, the program will adjust the priority sequence based on a default goal list as shown in Figure 3. The goal list suggests that designers need to handle range, counter, and sink before dishwasher, and finally arrange the location of the refrigerator.

If the user wants to process the design without checking for conflict, then the program will ask the user to locate the starting point of the first design unit. Afterward, the program will follow the priority list (for instance, refrigerator-sink-range-counter-dishwasher), spatial relationships among units, and the location of openings to process the design and to draw the output as shown in the left image of Figure 4. If the priority list is refrigerator, sink, range, counter, dishwasher, and users want to put the entrance door on the lower left corner, then the program will define the constraint to let the refrigerator face the opening first, then follow the priority list and the spatial relationships to arrange the layout. Results are shown in the right image of Figure 4.

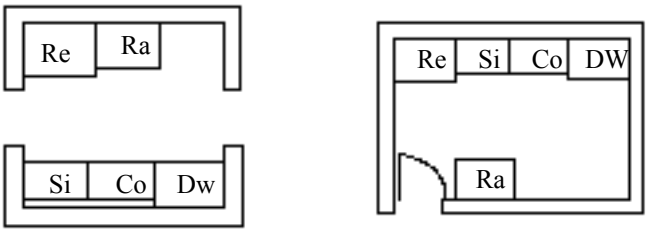


Figure 4. The kitchen design without and with the opening constraint.

2.3. EVALUATION OF THE SYSTEM

This system has a limited set of rules and well-defined steps, thus it generates limited results which are not flexible enough to show diversity. In order to generate more diversified design results to enrich the design quality and to involve more aspects of design thinking, a different approach was tested in the

following by manipulating various design rules.

3. Example two: Puzzling design issues

Another notion regards architects as puzzle makers instead of problem solvers. In this regard, designers are looking for and "*seeking unique sets of combinatorial rules (the essence of the puzzle) that will result in an internally consistent fit between a specific kit of parts (architectural elements, attributes or ideas) and the effects that are achieved when those parts are assembled in a certain way*" (Archea, 1987 p. 41). The following simulation program adjusts this concept by creating a set of rules simulating design intuition. This program was written in AutoLISP and executed in AutoCAD. A user interaction is developed to determine what the design units should be and what the spatial relationships are. A foyer design of a suburban house is simulated in this second study.

3.1. GENERAL METHODS

This simulation concentrates on designing a foyer in a suburban house. The suburban housing types and how they are represented and described in "plan books" are studied first. The suburban house type was chosen for its generality, familiarity, and conventional typology. The strong typology in suburban house design allows the abstraction of the design process to be represented logically and implemented digitally. In order to have a digital design tool, a representation of the design process is needed. Therefore, based on the elements constituting the space of a foyer found in the plan book, a priority list is set up to signify the process. This list has (in order of priority) a door, a staircase, a closet, a table, and space to display photos.

The execution of the program is mainly determined by generating variables specifying the priority of design elements through a series of questions. The program will gather user input by asking questions that are closed ended. The arranged questions relate to the social and psychological aspects of the identity of a foyer and how the foyer is used. Each question contains two or three possible answers from which to choose. Answers provided by the user will serve as a data source for the program to establish its knowledge base. An external file recording the user's responses is then generated. These "user data" answer questions about why things are the way they are. It also provide users with understanding about the character of the foyer space. This file is a record of the design variables that contributed to the final product. The uniqueness of the space designed depends upon how the designer answers the questions. Thus, the questions represent rules for determining the location of the objects, and the system is rule based.

3.2 APPLIED RULES

The first element needed is a door. Without a door, the name “foyer” does not apply. The options for door placement are type and location. The two available types are right handed or left handed, and the placement of the door is at the midpoint of a wall. The wall used is either the long or short side of the space created by the user. Therefore, the question of "Are you <L>eft or <R>ight handed?" determines the door type.

The second question, "Are you alone?," determines on which wall to place the door. If the user answers “yes,” the door is placed on the short side; otherwise, it is on the long side. After the door is placed, the next unit is the staircase. There are two types of stairs: a spiral stair (three sizes) and a straight stair. The placement of the stair is determined by the following questions. The first subquestion, "Do you want a two-story house?" sets up the type of single floor or double floors. If the answer is “yes,” then a stair will be placed in the foyer. The question "Are you in a hurry?" determines the type of stair. A straight stair is inserted for “yes” and a spiral stair is inserted for “no.” The questions "Do you sit on stairs?" and "Do you wander in your home?" are used to determine the size of the spiral stair. A 72"-diameter spiral stair is used for both answers of "yes" and a 48"-diameter stair for both answers of "no." Any other combination of responses of “yes-no” or “no-yes” will have a 60"-diameter stair.

The question "Does it matter to you that your neighbor outside might see you in your underwear?" further defines the straight-stair option. If the user answers “yes,” then the stair is placed parallel to the entrance, and if the user answers “no,” then the stair is placed perpendicularly. The placement of the staircase is in relationship to the first element, the door. The program uses a combination of “if-then” clauses to reference its knowledge base and places the door based on the decisions made by the user. The algorithms for the placement of a staircase are based on assumptions that, if the user does not entertain, a stair can occupy the internal space of the foyer. This means the stair is placed within the foyer instead of adjacent to it.

The next element is a closet. Its size is decided by the question, "When traveling overseas for two weeks would you bring a <T>oothbrush, a <B>ackpack, or a <S>uitcase?" Its placement is dependent on the door type and location. A closet will be placed adjacent to the foyer walls oriented to face the opening of the door. This rule follows the placement scheme illustrated in the plan books.

The next unit is a table whose size is determined by the question, "In a given year do you receive a <S>mall, <M>edium, or <L>arge number of gifts?" The placement of the table is in relation to the stair, assuming that the foyer is used as a temporary depository of "stuff." The more stuff received, the larger the table is.

01. Are you <L>eft or <R>ight handed?	→ "Right"
02. Do you want a two-story house? <Y>es or <N>o:	→ "Yes"
03. Do you sit on stairs? <Y>es or <N>o:	→ "Yes"
04. Do you wander in your home? <Y>es or <N>o:	→ "Yes"
05. Are you in a hurry? <Y>es or <N>o:	→ "No"
06. When traveling overseas for two weeks would you bring a <T>oothbrush, a <B>ackpack, or a <S>uitcase?	→ "Backpack"
07. In a given year do you receive a <S>mall, <M>edium, or <L>arge number of gifts?	→ "S"
08. Does it matter to you that your neighbor outside might see you in your underwear? <Y>es or <N>o:	→ "No"
09. Are you alone? <Y>es or <N>o:	→ "No"
10. Would you like any of the things listed above in your foyer? <Y>es or <N>o:	→ "Yes"
11. Would you like to suggest another item? <Y>es or <N>o:	→ nil
12. Please name new object:	→ nil
13. Should I place a stair here? <Y>es or <N>o:	→ "Yes"
14. Do you want a table? <Y>es or <N>o:	→ "No"
15. Do you want a place to display photos? <Y>es or <N>o:	→ "No"
16. Would you like a closet here? <Y>es or <N>o:	→ "Yes"

Figure 5. Example of a user input data base and its data file.

The following questions determine what a user wants inside the foyer: Should I place a stair here? Do you want a table? Do you want a place to display photos? Would you like a closet here? After answers are collected, the program starts to generate the foyer. The results then are displayed in ACAD and answers are collected on a data file. The three-dimensional model also will be shown in 3DSMAX 2.5 software to allow real-time animation of the generated design result. Shown in Figures 5 is an example of a set of recorded user data, and its generated design is shown in Figure 6.

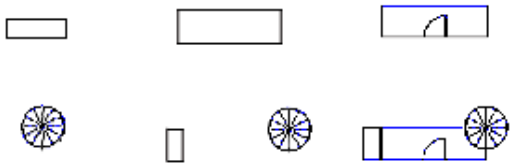


Figure 6. Result of the generated design.

4. Conclusions

If a model is used to represent an observed reality, it should be capable of depicting a believable version of that reality. In the process of producing a design, many factors and variables are involved, and it is difficult to locate and identify all of them because design is a mental activity that occurs in the designers' mind. It is impossible to include all variables of a design process in a finite model. Therefore, the criteria for judging a good design model are not

based on how well it mimics a human designer, but whether it can generate reasonable designs. There are both advantages and disadvantages to the two simulation models. The first model is rigid and logically oriented. To make it flexible, the program should include more reasoning and algorithms to handle different situations other than the kitchen triangle. In other words, the knowledge base should be enlarged to include a broader range of constraints.

The second example has more design intuition involved. However, the setting of the design rules is subjective and arbitrary. They represent only one pattern of a design process obtained from the analysis of the plan book. To create more possibilities, the system should ask more questions to obtain more data for generating a larger knowledge base to make the design interesting. Of course, more questions might generate conflicts between the user data and the program's decision-making process. Also, a suburban house may have a standard layout which limits the range of styles. Other building types, however, should benefit from a larger knowledge base. In sum, it is possible to combine both methods of logic and intuition together to create a more promising design. It also hopes that the information yielded and methods provided in this study stimulates future research on design automation.

References

Archea, J.: 1987, Puzzle-making: What architects do when no one is looking, *in* Kalay Y. E. (ed.), *Computability of design*, John Wiley & Sons, New York, pp. 37-52

Eastman, C. M.: 1973, Automated space planning. *Artificial Intelligence,* **4**, 41-64

Kolodner, J.: 1993, *Case-Based Reasoning*, Morgan Kaufman, Inc, CA

Newfert E.: 1980, *Architects' Data*. Crosby Lockwood Staples, London

Newell A. Shaw J. C. and Simon H. A.: 1962, The process of creative thinking, *in* H. E. Gruber H. E. and Wertheimer M. (eds.), *Contemporary Approaches to Creative Thinking*, Atherton Press, New York, pp. 63-119

Newell A. and Simon H. A.: 1972, *Human Problem Solving*, Prentice-Hall, Englewood Cliffs, NJ

Schon, D. A.: 1983, *The Reflective Practitioner: How Professionals Think in Action*, Basic Books, New York

Stiny C. and March L.: 1981, Design machines, *Environment and Planning B*, **8**, 245-255.