Constantin Terzides and Emmanuel-George Vakal—

College of Architecture and Urban Planning, The University of Michigan,
Ann Arbor, MI 48109-2069 U.S.A. Tel.: 313.763.2077 Fax: 313.763.2322

SOME THOUGHTS ON THE ROLE OF COMPUTERS IN ARCHITECTURAL DESIGN

Abstract: This paper attempts to identify, describe, and evaluate the role of computers in the architectural design process. The role of the computer in the design process over the last thirty years is viewed as having evolved through three stages: automated, augmented, and formalistic. Some possible future roles computers can play in architectural design are outlined.

1. THE DESIGN PROCESS

To identify the problem of design in general, and of architectural design in particular, it is necessary to describe and understand the *process* of design. While many definitions and models of design exist, most agree that "design is a process of inventing physical things which display new physical order, organization, form, in response to function" [Alexander, 1964: 6]. However, since no formula or predetermined steps exist which can translate form and function into a new, internally-consistent physical entity, design has been held to be an art rather than a science. It is considered to be an iterative, "trial-and-error" process that relies heavily on knowledge, experience, and intuition.

Intuition became a basis of many design theories, often referred to as "black box" theories. According to them, design, as well as its evaluation, tend to be highly subjective. In contrast, another set of theories define the design process as a *problem-solving process*. According to the latter, design can be conceived as a systematic, boundedly, rational activity. As defined by researchers like Alexander [1964], Newell and Simon [1972] over the past twenty years, for every problem there exists a solution space, that is, a domain that includes all the possible solutions to a problem. Problem-solving then can be characterized as a process of searching through alternative solutions in this space to discover one or several which meet certain goals and may, therefore, be considered *solution states*. The way by which the design problem will be solved can be either deterministic or probabilistic.

The objective of this paper is to identify, describe, and evaluate the role of computers in architectural design. Automated design systems are discussed first, with particular emphasis on expert systems and artificial intelligence. Augmented design systems are discussed next; their inherent difficulties are pointed out in an attempt to explain the reasons for their marginal contribution in the architectural design process. Third, a new concept, that of formalistic design, is discussed briefly. Recent developments in formalistic design are presented.

2. AUTOMATED DESIGN

In the early 1960s, Alexander [1964] published a highly influential book titled *Notes on the Synthesis of Form* . In it Alexander quotes the need for rationality in the design process. If design, he argues, is a conceptual interaction between the context's demands and the inadequacies of the form, there may be a way to improve it by making an abstract picture of the problem, which will retain only its abstract structural features. As a mathematician, he introduced set theory, structural analysis, and the theory of algorithms as tools for addressing the design problem. Quality issues can be represented by binary variables. If a misfit occurs, the variable takes the value 1; if not, 0. Each

binary variable stands for one possible kind of misfit between form and context. This approach was followed by a flurry of related research into the problem. However, Alexander's contribution was much more far-reaching. He introduced computers into the design process by suggesting which aspects of the design process are amenable to systematization and which are not. Further, he suggested that the design process entails frequent changes of mind (or changes of constraints, in scientific terms) and that a system should permit these changes to occur. One of the areas where the computer can be helpful to an architect is in space allocation, in finding a large number of possible schemes at a sufficiently early stage of the design process, and choosing the best one for further development. An early attempt was MIT's BUILD system [Dietz, 1974] which could be used to describe spaces that might go into a building, indicating their dimensions, their arrangement, and their materials. The computer then arranged the spaces.

However, because of the large number of constraints to be simultaneously considered in an architectural design problem, it would be difficult to meet them all. Moreover, the complexity of the design problem is so great that a designer would be unable to arrive at an appropriate solution unless a new way could be found to break down the problem into sub-problems and use a non-deterministic approach to solve them [Bernholtz, 1969].

Another approach to solving design problems is that of *linguistics*. Here, the designer attempts to structure the problem by grouping the constraints into thematic areas (e.g. zoning, circulation), and proceeding to design by considering each group of constraints more or less independently. This information is converted into linguistic structures through the use of transformational rules. Then, the designer represents these linguistic structures in the form of sentences [Chomsky, 1957], that is, specific sets of architectural elements which include not only the elements but also the rules which allow a designer to combine them into feasible and meaningful architectural compositions. The aim of this approach became one of writing algorithms for the generation of feasible and meaningful architectural "sentences."

Although not directly related to linguistics, the theory of algorithms was worked out in considerable detail by Markov. He suggests that any algorithm should be:

-definitive, and universally comprehensive
-general
-conclusive

Algorithms have been written for the design of buildings, or rather, for designing parts of buildings. Over half of these algorithms have been concerned with *space allocation* problems, some of which resulted in formal definitions or languages which the computer can be programmed to resolve, such as SIPLAN [Yessios, 1975].

Some theorists have argued that many problems cannot be solved algorithmically [Gill, 1978], either because the procedure leading to their solution is ill-defined or because not all the information needed to solve them is available or accurate. Such problems make it necessary to use *heuristic and adaptive decision procedures*. Heuristic methods typically rely on trial-and-error techniques to arrive at a solution. Such techniques are, by definition, much closer to the *search-and-evaluate* processes used in architectural design. In adaptive procedures, the computer itself learns by experience, as in Negroponte's "architecture machine" [1970], which could follow a procedure and, at the same time, could "discern and assimilate" conversational idiosyncrasies. This machine, after observing a user's behavior, could reinforce the dialogue by using a predictive model to respond in a manner consistent with personal behavior and idiosyncrasies. The dialogue would be so intimate, "that only mutual persuasion and compromise would bring about ideas." [Negroponte, 1970: 13] The role of the machine would be that of a close and wise friend assisting in the design process.

This approach became the basis of relatively recent developments. In systems, known as *expert systems* , knowledge about a specific area of human expertise is codified as a set of rules. By means of dialogue with the user, the system arrives at a solution to a particular problem. New knowledge is provided by the user to the knowledge base without a programmer having to rewrite or reconfigure

the system. The ability of the system to justify conclusions and to explain reasoning, leads to further systematization of the design process, but also, sometimes, to unpredictable behavior by the computer.

3. AUGMENTED DESIGN

As a result of growing computer capabilities during the 1960s, automated design engendered a great number of expectations. Unfortunately, most of these expectations were not met, perhaps because machine intelligence was overestimated. Architectural design is a much more complicated process than many other processes because it entails factors that cannot be codified or predicted. The heuristic processes that guide the search rely not only on information pertinent to the particular problem, but also on information which is indirectly related to it. In addition, the states that describe the design process do not exist before they are generated. Therefore, a solution state can only be identified "after the fact", that is, after it has been generated.

These problems, as well as the computer needs of architectural offices, led to changes in the approach to Computer Aided Architectural Design (CAAD). Rather than emulating architects, the approach in the 1970s was predicated on the belief that they should be eliminated. The machine was introduced as an aid to instruction, as a mediator for the goals and aspirations of the architects. The computer could communicate with architects by accepting information, manipulating it, and providing useful output. In addition to synthesizing form, computers are also able to accept and process non-geometric information about form. Therefore, it is necessary for architectural design languages to be invented to describe operations on building databases. One pioneering effort in this area is GLIDE [Eastman and Henrion, 1976], a language which allowed the user to assemble buildings.

Another approach in the direction of computer-augmented architectural design, was the manipulation of architectural forms according to rules [Mitchell, 1974]. Basic structural and functional elements were assembled to make volumes (elements of composition) which, in turn, were assembled to make buildings. All elements were stored in the computer's memory in symbolic form, and the user operated on them by manipulating symbols in accordance with rules derived through the classic academic tradition.

Negroponte's [1974] early vision of machine intelligence, came true with a pioneering system which allowed the user to generate a design solution by employing elements from a standard menu displayed on the screen.

As design began to be increasingly thought of as a systematic and rational activity, many of its empirical and experimental rules were explored. By operating on symbolic structures stored in the computer's memory and manipulating them according to rules, computers could reason about, or even predict, the behavior of a simulated environment. The machines were made to carry out a "make-believe" happening, a *simulation*.

Numerous simulation models were formulated and much progress was made toward simulating design states [Rasdorf and Kutay 1982; Lafue 1979]. These models simulated the states of a designed environment and the transitions from one state to another. Yet, no model was formulated which could encompass both the relationships between the components of a building and its environment.

Even though simulation models are valuable tools for predicting and evaluating performance, their contribution to the architectural design process has been marginal. They leave the interpretation of the symbols they represent and the relationships between them to the designer. The transition from one design state to the next must be done by the designer with little or no assistance by the computer.

Augmented design failed to improve the architectural design process and products. Over 90 percent

of the systems that have been installed worldwide are used for drafting or site planning, which are not, in themselves, essential steps in the process of architectural design. [Leighton, 1984] Perhaps in order to employ them in the design process more effectively, the utility of computers in simulating design states must be extended beyond descriptive geometric or non-geometric information. It must also include the semantics (meaning) of that information. [Kalay, 1985]

4. FORMALISTIC DESIGN

So far the architectural design process has been examined from a problem-solving point of view and various attempts to satisfy design criteria have been presented from a historical perspective. These attempts have fostered a number of visions. Unfortunately, most of these visions remain just that and the enthusiasm for computer-aided design of the 1960s became skepticism.

The failure of CAD to improve the architectural design process and products is probably due to the fact that most of the researchers did not consider the idiosyncrasies of architectural design. In architecture, design quality is reflected in forms and their relationships. Many architects and theorists have argued that what distinguishes a well-designed building from one that is poorly designed can only be found in the morphological relations that the former embodies. "One can have a beautiful idea of winning a chess game. One can brutally win a chess game in a very inelegant way. But there can be an elegance in the process of winning itself, that is poetic. We are looking for the poetic in the process, regardless of the result. We are looking for a beauty internal to the idea of the play, that is, when one suddenly gets the shiver." [Ford, 1986: 34]

Formalistic design is viewed as an activity, which entails invention and exploration of new forms and their relations. Various methods of analysis have been employed in the search of new forms: formal analysis involves the investigation of the properties of an architectural subject. Composition, geometrical attributes, and morphological properties obeying Galilean and Newtonian principles are extracted from figural appearances of an object. In contrast, structural analysis, deals with the derivation of the motivations and propensities which are implicit within form and which may be used to define the limit between what it is and all other possibilities. [Brown,1986]

The formalistic approach to computer-aided architectural design is relatively new. Therefore, the literature in the area is quite limited.

One approach to formalistic design is that of *shape grammars* [Stiny 1985; Flemming 1986]. They were developed to carry out spatial computations visually and are used to generate designs based on formal rules. A shape grammar consists of rules and an initial shape. There are two types of shape grammars. In standard grammars, each rule is defined explicitly by a pair of shapes separated by an arrow. The shape on the left side of the arrow determines the part of the shape to which the rule is to be applied. The shape on the right side of the arrow determines the shape that results when the rule is employed. In parametric grammars, rules of this kind are defined implicitly by rule schemata. These allow the lengths of lines and the angles between lines to be altered. Shape grammars reveal a lot about languages of design. They have been used extensively for the generation of patterns, diagrams, and floor layouts.

An interesting variation of shape grammars is that of *fractal generative systems* . Based on a scheme, formulated by the German mathematician Von Koch, a fractal process, consists of an initial shape (the base) and one or more generators. From a practical point of view, the generator is a production rule: each and every line segment of the base is replaced by the shape of the generator. The implementation of an interactive computer program has been reported by Yessios [1987] which allows the fractal to be generated one at a time or at multiple increments, backwards or forwards. As described by Yessios, "a building typically has to respond to a multiplicity of processes, superimposed or interwoven. Therefore, the fractal process has to be guided, to be constrained and to be filtered. The fractal process has to be 'mutated' by the utilitarian requirements of the functionalities of a building." [Yessios, 1987: 7]

Another approach to formalistic design is that of *transformations*. It involves two important principles of architectural form: stability and change [Eisenman, 1986]. A transformation is not exactly a form-making procedure because the subject of transformation must already be complete. In a transformation only relations change. No new elements can be introduced or removed; bits cannot be added or taken away. However, the illusion of movement, often described as "frozen movement", has been argued to have a high architectonic value. [Evans, 1986] It illustrates the forces designers have referred to, as "punctured volumes," "compressed planes," "interpenetrating spaces," or "agitated surfaces."

The concept of transformation has not yet been implemented extensively in computer-aided design. One interesting exploration of *shape transitions* has been reported by Yessios [1987]. According to him an initial shape A can be transformed to a target shape B by applying any number of in-between steps. All the points of shape A are mapped onto shape B and vice-versa. Furthermore, once the rules of transition have been established, the transition can be allowed to continue beyond its target, to infinity.

As opposed to automated or augmented design, formalistic design has the advantage of allowing one to delve into the idiosyncrasies of architectural form. Its disadvantage is that it is perceived as being too iconolatric, superficial, and conformist. [Calinescu, 1987] As a consequence, formalistic design has always been regarded suspiciously as combining the icons of historical architecture and technological development at a surface level.

5. THE FUTURE OF COMPUTER-AIDED ARCHITECTURAL DESIGN

There is often a conflict between application, where the tendency is to get a job done, and theory, where the tendency is to ponder why and how things are done. It is true that one cannot develop theories forever about the way ahead: one must eventually "walk" down that road. On the way, one should consider his goals and how to reach them.

Earlier in this paper, the role of computers in architectural design was examined from an historical perspective. In the foregoing paragraphs, the future of computer-aided architectural design is discussed. The following sections are based on early visions of the roles computers can play in architectural design.

“Computers are intellectual machines that allow us to simulate human behavior." [Negroponte, 1970: 1]

In developing computer programs one is forced to question how people think and how designs evolve. In other words, computers must be acknowledged not only as machines for imitating what is understood, but also as vehicles for exploring what is not understood. The entire sequence of specifying computer operations is similar to that of human thinking. When designing software for natural language understanding, knowledge representation, inference, or learning, one is actually transferring to a machine processes of human thinking. The computer becomes a mirror of the human mind, and as such, it reflects its thinking. Therefore, design can be explored as mental process not only by observing human behavior, but also by observing the machine’s behavior. To do this, it is necessary to perform individual operations with substantial independence; that is, the entire sequence of operations must be such that there is no human intervention from the time data is entered until the results are obtained and that design decision-making mechanisms be built into the machine itself. This does not mean that a “computer-designer” is to be created even though that may be desirable eventually. Rather, it suggests the attainment of independence in solving particular design problems. Thus, the designer can observe via the computer his decision-making process and compare it with that of others.

"A world view of a culture is limited by the structure of the language which that culture uses." (Sapir and Whorf hypothesis)

At present machines have denatured language-codes. To employ computers more effectively in the design process, databases must extend beyond mere geometric and non-geometric codicil information. They must also include the meaning of that information. For example, in architectural design, software should allow recognition of a square regardless of its size and orientation. Today, one can begin to see systems allow one to perform a large number of calculations and are moving towards larger databases that permit interaction. In the future, databases may include information on the meaning of objects, events, and relations thereof.

"It is not very difficult to make machines that will play chess of a sort. The mere obedience to the laws of the game, so that only legal moves are made, is easy within the power of quite simple computing machines." [Wiener, 1947: 171]

Games employ rules, strategies, tactics, and goal-seeking that may be useful beyond the game's boundaries, e.g., in design. However, it is unclear whether it is possible to construct a game-playing system that can use design rules and whether this capability would represent an essential difference between the potentialities of the machine and those of the mind.

There is a theory of games (von Neumann, 1945) which establishes a way to describe and analyze them and their strategies by working from the end of a game rather from its beginning. In the last move of the game, a player strives to make a winning move if possible, or, if not, at least a drawing move. When the entire strategy is known, this is manifestly the best strategy for playing the game. However, in design, available knowledge is not sufficient to permit the formulation of a complete strategy of this sort. In design, strategies can only be approximated. Such a strategy would rely on the relation of local actions to global intents, on attitudes not always justified. For example, the local moves embodied in construction procedures are characterized by specificity. In contrast, global goals can be quite ambiguous.

Given that the global intentions of architectural design are, at best, ambiguous it is not necessary to question whether it is possible to construct a game-playing machine which will generate an optimum solution by following rules in von Neumann's sense. On the contrary, it is unquestionably possible to construct systems that undertake local actions irrespective of the final goal. The real problem is to construct a system which will offer opportunities for interesting and challenging dialogue with a designer. Such a system would be capable of learning by experience which will enable it to improve its knowledge of the strategy and rules of architectural design. The point of departure for this learning would be a set of rules derived statistically from architectural precedent.

6. CONCLUSION

“Let us build machines that can learn, can grope, and can fumble, machines that will be architecture partners, architecture machines." [Negroponte, 1970: 121]

"The aim has been to leave the value judgement to the user and have the system do all the procedural and tedious work which will check the feasibility of the user's propositions " [Yessios, 1975: 3]

In the 1960s the role of computers in architecture was replicate of human endeavors and to take the place of humans in the design process. In the 1970s the role was to create systems that would be intelligent assistants to designers, relieving them from the need to perform the more trivial tasks and augmenting their decision-making capabilities. Today, computers are increasingly involved in the design process. Their roles vary from drafting and modelling to intelligent knowledge-based

processing of architectural information. While the future of computers appears to include a variety of possible roles, it is worth exploring these roles in the context provided by the question: “Who designs?” If one takes the position that designing is not exclusively a human activity and that ideas exist independently of human beings, then it would be possible to design a computer mechanism which relates ideas.

References

1. Alexander, C., *Notes on the Synthesis of Form,* Cambridge: Harvard University Press, 1967.

2. Berkeley, E., "Computers for Design and Design for the Computers," *Architectural Forum 128(2),* (March 1968).

3. Bernholtz, A. and Bierstone, E., "Computer Augmented Design," *Design Quarterly* , (1966/67).

4. Brown, A., "In Caesura," in *Eisenman Studios at GSD: 1983-85,* Cambridge: Harvard University Graduate School of Design, 1986, p. 35.

5. Calinescu, M., *The Five Faces of Modernity,* Bloomington: Duke University Press, 1987.

6. Chomsky , N., *Syntactic Structures,* The Hague: Mouton and Company, 1957.

7. Dietz, A., *Dwelling House Construction,* Cambridge: MIT Press, 1974.

8. Eastman C. M. and Henrion M., *M. GLIDE: Language for a Design Information System,* Pittsburgh: Carnegie-Mellon University, Institute of Physical Planning, 1967.

9. Eisenman P., "The Futility of Objects," *Harvard Architecture Review 3* , (1984), p. 66.

10. Evans, R., "Not to be Used for Wrapping Purposes," *AAFiles 10*, (1987), p. 70.

11. Flemming, U., "The Role of Shape Grammars in the Analysis and Creation of Design," *Proceedings of Symposium on Computability of Design at SUNY Buffalo*, (December 1986).

12. Ford, K., "In Caesura", in *Eisenman Studios at GSD: 1983-85,* Cambridge: Harvard University Graduate School of Design, (1986), p. 35.

13. Gill, A., *System Modeling and Control,* New York: John Wiley and Sons, 1978.

14. Kalay, Y., "Redefining the Role of Computers in Architecture: from Drafting/Modelling Tools to Knowledge-Based Design Assistants," *Computer Aided Design 17(7)*, (1985).

15. Lafue, G. M. E., "Integrating Language and Database for CAD Applications," *Computer Aided Design 11(3)*, (1979).

16. Leighton, N., *Computers in the Architectural Office,* New York: Van Nostrand Reinhold, 1984.

17. Mitchell, W., "Vitruvius Computatus," in W.F.E. Preiser (ed.), *Proceedings of EDRA 4 Conference*, Stroudsbourg: Dowden, Hutchninson and Ross, 1973.

18. Negroponte, N., *The Architecture Machine,* Cambridge: MIT Press, 1970.

19. Negroponte, N., *Soft Architecture Machines,* Cambridge: MIT Press, 1974.

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

21. Rasdorf, W. J. and Kutay, A. R., "Maintenance of Integrity During Concurrent Access in a Building Design

Database." *Computer Aided Design 16(4)*, (1982).

22. Stiny, G., "Computing with Form and Meaning in Architecture," *Journal of Architectural Education 39,* 1985.

23. von Neumann, J. and Morgestern, O., *Theory of Games and Economic Behaviour*, Princeton: Princeton University Press, 1944.

24. Wiener, N., *Cybernetics or Control and Communication in the Animal and the Machine*, Cambridge: MIT Press, 1948.

25. Yessios, C., "A Fractal Studio," *ACADIA 87 Proceedings* , North Carolina State University, (1987).

26. Yessios, C., "Formal Languages for Site Planning," in C. M. Eastman (ed.), *Spatial Synthesis in Computer-Aided Building Design*, New York: Wiley, 1975.

27. Yessios, C., "Syntactic Structures for Site Planning," in W.F.E.Preiser (ed.), *Proceedings of the EDRA 4 Conference*, Stroudsbourg: Dowden, Hutchninson and Ross, 1973.