Strategies for Interactive Design Systems

by

Robert F. Woodbury

EDRC-48-02-87

September 1986

STRATEGIES FOR INTERACTIVE DESIGN SYSTEMS

by Robert F. Woodbury

September 26, 1986

CARNEGIE-MELLON UNIVERSITY

Department of Architecture
Pittsburgh, Pennsylvania, 15213
(412) 268-8853

Abstract

An information processing model of human problem solving is used to develop strategies for the design of systems for the interactive generation of designs. Systems of this type are currently not strongly developed anywhere, nor does there exist in the literature a paradigm for their creation. Design is a task which requires different interactive support than that traditionally provided by CAD systems. In this paper, those differences are uncovered by comparison of two tasks; one, named definition in this paper, which seems to be well supported by existing systems and the other, the task of design. Use of an information processing model of human problem solving shows that differences between the tasks can be found in every potentially variant portion of the model. The information processing model is again used as a framework to propose mechanisms to support design. These mechanisms act by changing the underlying phenomena upon which the information processing model is built and thus effecting changes, either parametric or structural, in the model. The relative importance of the proposed mechanisms is discussed, leading to the conclusion that the interactive support of search is the most strategic direction for future research.

THE MOTIVATION

This inquiry starts with the premise that a computer-aided media for interactive design by humans is a desirable goal to pursue. Its approach can be placed in the world of computer aids to design by referring to a classification of such aids originally by Galle [Galle 81] and augmented by Akin, Flemming and Woodbury [Akin 84].

1	**Automated evaluation of designs which are generated by traditional means**
2	**Interactive generation of designs using computer-assisted media**
3	**Stepwise interactive generation[1]**
4	**Non-exhaustive generation of feasible solutions**
5	**Exhaustive generation (i.e. complete enumeration) of feasible solutions**
6	**Generation of (sub)optimal designs**

This work belongs to the second category, of interactive generation of designs, using computer assisted media. Its approach is based upon the belief that the methods which humans use to design are intellectually challenging and exciting; in short they are fun. The joy of discovery that comes with a successful design idea and the sense of playfulness seen in the work of many good designers are among the things that should be supported and aided by a high quality interactive system. That no such systems exist today is a testament to the difficulty of the problem.

It seems obvious that computer systems, as responsive assistants to design, potentially have a tremendous advantage over purely passive media such as pencil and paper. However, current modelling systems do not seem to live up to this promise. They are instead rather good means to define something which has already been created, either on paper or in the physical world [Woodbury 85]. In spite of much interest, beginning over twenty years ago with SketchPad [Sutherland 63], current systems do not seem to be natural media for aiding the design and development of objects. Yet designers create and manipulate geometric representations of objects and even of things that are not physical objects (for example, a scheduling chart) continuously in the course of design. This distinction, between the capabilities of current systems for definition of form and the desired functionality of future systems to act as a medium for the design of form is the topic addressed here. This paper is meant to define a strategy for the development of interactive systems which can function as media for design. Design refers to a wide range of activities, from an individual effort at sketching plans for a piece of furniture to the workings of an interdisciplinary team creating plans for a large building complex. This paper addresses only a very small part of that range; design is viewed here as being done by one person, working with a medium on which representations of design are recorded.

---

[1] In stepwise interactive generation, design is viewed as a sequence of transformations upon some representation. At any stage in design a set of currently feasible tranformations is generated automatically. A designer guides the process of design by making a selection from that set

DESIGN VS DEFINE

If, as stated above, design is a different form of activity than defining something which already exists, then the two acts of definition of an object and design of an object need to be distinguished. The nature of the activity of design has been widely discussed in the literature [Simon 73] [Rcitman 64]. There is still much to be learned of the phenomenon called design, particularly in its higher level, problem restructuring, stages. The exceptions to this literature notwithstanding, for the purposes of this discussion, let design be described as:

"the creation of a representation of an object which meets a set of requirements[9]*

Let definition (also referred to as *define* in this paper) be described as:

"the creation of a representation of an object given either the object itself or an informationally complete alternative representation of the object[2]"

An example of the design task is the design of a bathroom layout, given a list of client requirements and constraints, which may include cost, number of fixtures, space allocated, lighting and others [Eastman 70], An example of the definition task would be the reproduction of a representation of the MBB Gehause part, a mechanical part often used in the literature as a benchmark of geometric modelling system performance.

Both of these acts, design and define, can be thought of as problems in the sense that they provide some given information and a task to perform with that information. For definition, the given information is an informationally complete representation of an object and the task is the creation of a model of that object. For design, the given information is more vague: a set of requirements on an object, and the task is the creation of a model of an object which fulfills those requirements. Further comparison of the two problems can be made by representing them in a conceptual framework for computer-augmented problem solving.

Newell and Simon [Newell 72] propose a theory of human problem solving that provides the basis for such a framework. The theory states that human problem solving can be considered as:

- **an *information processing system* (IPS)**
- **using a *problem space***
- **acting by *searching***
- **influenced4>y a *task environment***
- **and supported by an *external memory.***

As these terms may be unfamiliar to readers of this paper, a brief discussion is included here. For a complete presentation of the theory, refer to Newell and Simon [Newell 72], particularly chapter 14.

---

2 *For* example, a set of drawings or a physical model

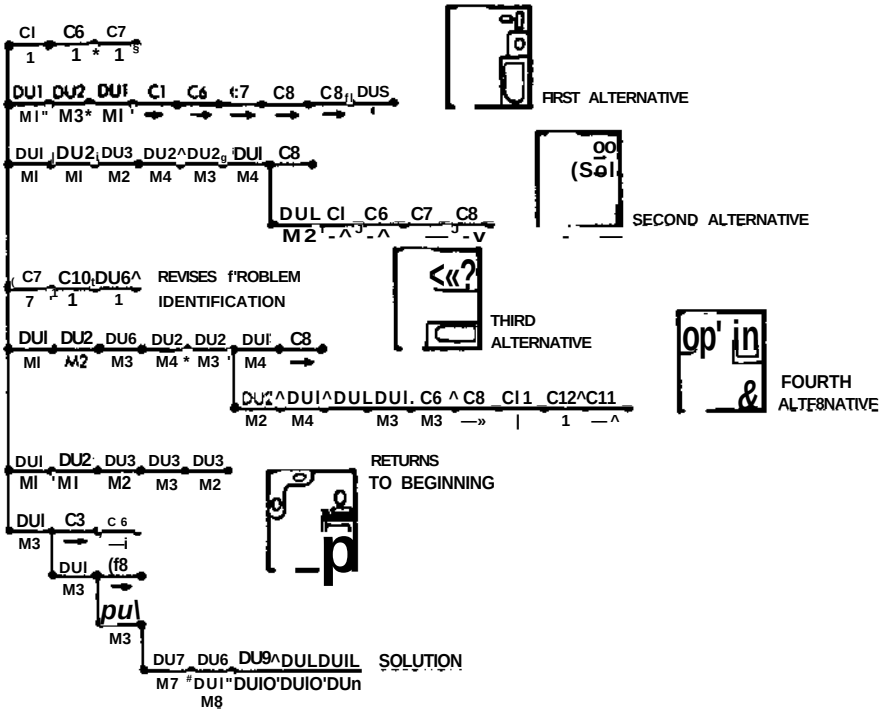


Figure 1: An example of a bathroom layout problem from [Eastman 70]

Human Problem Solving

The human information processing system (IPS) can be modelled as a number of memories acted upon by processors, which gain information from sensory devices and make physical actions in the real world through motor devices (see figure 3, after Card,Moran and Newell [Card 83]). Each of these memories and processors have performances, the parameters of which are largely invariant across individuals. The inherent structure (or architecture) of the IPS along with these parameters dictate significant performance characteristics of the IPS as a whole which affect the capability of the IPS to engage in problem solving. For example, short term memory (STM) capacity in the human IPS is very small. This, coupled with a long write time to long term memory (LTM) ensures that only problem solving strategies which have small stocks of states are used by an unaided IPS.

Newell and Simon argue that the characteristics of the human information processing system to a large degree guarantee that the human IPS will only engage in certain classes of problem solving behavior. The behavior universally observed in typical problem solving can be characterized as "search in a problem space". Human problem solvers act by considering internal symbol structures (elements in a problem space) until a solution is found. The constituent components of a problem space are, (after Newell and Simon [Newell 72]):

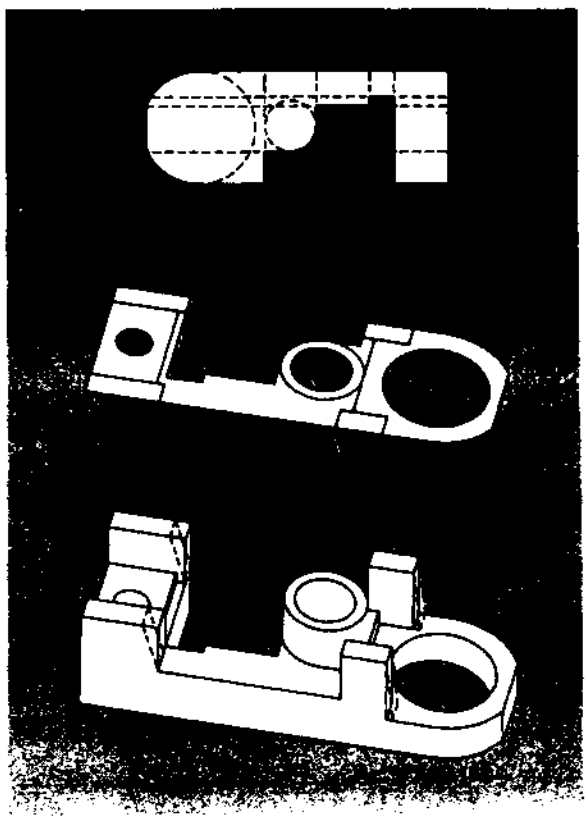


Figure 2: Definition sequence for the MBB Gehause part

1. a set of symbol structures, each representing a knowledge state. These symbol structures need not exist explicitly in memory prior to generation. All that need exist is a means to generate the symbol structure through some operator.

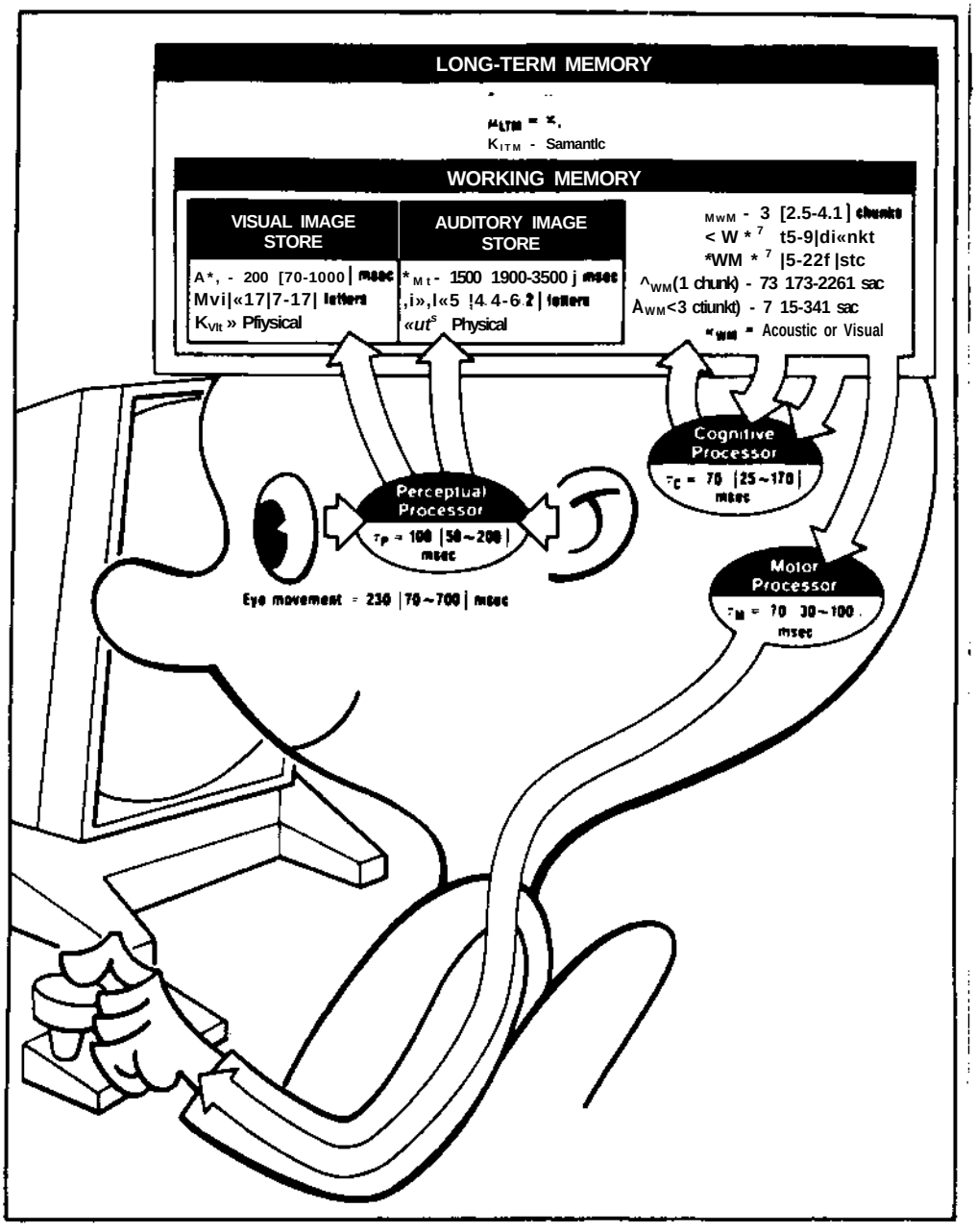
2. a set of operators to incrementally generate new states of knowledge or to make a transition between states of knowledge.

3. an initial state of knowledge.

4. a problem, which is a description either complete or very vague of a final, desired state of knowledge.

5. a set of knowledge. Part of this knowledge may exist outside of the physical body of **the** problem solver in the form of external memory (EM). Examples of such memory **are** drawings, books, chessboards and computing devices.

Newell and Simon report for their subjects that the use of a problem space was invariant across subjects and tasks. They do not exclude the possibility of other environments for problem solving **but** also find no evidence for any such alternate environment. Simon [Simon 73] further points out that this formalism can be used to explain complex, ill-structured domains such as design.



li - the storage capacity in items
8 - the decay time of §n item
K - the main code type (physical, acoustic, visual, semantic)
r - the cycle time

Figure 3: The architecture of the human information processing system (IPS)

Problem solving occurs through a process of search in some problem space. Search involves visiting states of knowledge in the problem space in some order, according to some control structure until some goal is met Search is thus an iterative activity, of successive application of operators to transform knowledge states. Four types of decisions must be made at every iteration in any search procedure:

1. **Evaluation of existing states of knowledge against the problem requirements.**

2. **Selection of an existing state of knowledge on which to operate.**

3. **Selection of the appropriate operator to apply to the selected state of knowledge.**

4. **Selection of existing knowledge states to remember.**

A set of policies, one for each of these types of decisions determines a search strategy. Search strategies take advantage of information in the problem space and are limited by the invariant characteristics of the human IPS. An example of search which utilizes information in the problem space is found in tic-tac-toe in which the search can take advantage of the symmetry of the game to reduce the number of existing states which are candidates for operation. An example of the characteristics of the human IPS constraining search is the use of depth first search for finding files in a hierarchical directory system [Akin 83]. Depth first search discards all but a small number of knowledge states. It is thus a useful search strategy for the human IPS with its limited capacity for short term memory (STM).

A task environment for problem solving is the reality in which problem solving occurs. A problem is posed with reference to some state in the world, but is solved in the internal environment of a problem space. For example, a problem to solve the Rubik's cube is solved internally and symbolically and the actual environment specified by the Rubik's cube serves to provide knowledge about the world mostly in the form of constraints (and in this case also acts as an external memory device to aid[3] problem solving). Newell and Simon report that the constraints in the task environment are the main determinants of the problem space used by a problem solver. The task environment may also provide information which determines the type of search used in problem solving. Thus a particular task further scopes the behavior of a problem solver.

Human problem solvers are often aided by the use of an external memory to augment other memories. External memory may be immediately available to a problem solver (the area of a piece of paper in direct foveal view) or may be more remote (a book in a library). Traditionally, external memory is perceived as a passive recipient of actions made by the problem solver. It is often the medium in which a solution to a problem is described. The external memory, being outside of the human organism is the component of human problem solving most amenable to change through a computer assist

Taken together, these five invariants, the human information processing system, the use of a problem space, the activity of search, the constraints of particular problem solving tasks and the existence of an external memory device* provide a framework for understanding a given problem solving domain.

Task Comparison

An attempt to compare both of these tasks within the framework given above immediately implies a question of fit. Are both tasks representable within the framework? Given that they are both representable, then the differences between the tasks will appear not as structural differences within the problem solving framework, but rather as differences within each element of the framework. Newell and Simon [Newell 72] claim that their theory, while based upon observation of three types of

---

[3] or in Rubik's case, seemingly to hinder

problem solving tasks (cryptarithmentic, logic and chess) is of broad generality. They argue that the structure of the human IPS is invariant across individuals and that that structure dictates the use of a problem space in problem solving. It should be expected then, that both the design and define tasks, if attempted by a human, will be solved within a problem space. The other potentially variant characteristic of human problem solving between these two tasks is search. Newell & Simon outlined a (fuzzy) distinction between "unprogrammed" activities involving search and "programmed" activities involving execution of a predefined script. Both types of activity are, in a sense, problem solving, as they involve transformation of an initial set of conditions into a final set of conditions. However, we tend to observe greater use of "programmed" activities as more "mechanical", and less typical of human beings acting to solve a tough problem. Even if one of the tasks studied here was typically solved by completely "programmed" or mechanical means, then it would still be sufficiently similar to the activity of problem solving to be discussed within the Newell & Simon framework. The distinction between the two tasks would then found be in the search part of the framework, in which the activity of the highly "programmed" means could be described as non-backtracking and non-iterative compared to the methods of search used for other less "programmed" problems. The actual representations and search strategies used by problem solvers will differ between the two tasks; the degree of that difference will indicate, to a large degree, the differences between the two tasks from a problem solving point of view. Given the invariance of the human IPS and the different types of search and representation that can be discussed within the bounds of the framework, it seems reasonable to compare both tasks within the framework.

Using this framework of problem solving allows comparison of the design task with the definition task and further understanding of their differences and similarities. In both cases, the human IPS remains invariant. Nothing, anywhere, suggests that normal, healthy people have information processing architectures which differ structurally or significantly parametrically from the one outlined above. That leaves the task environment, the problem space and the search strategy as the domains in which to seek comparisons. In the absence of empirical data, I will rely on argument to make the comparisons.

The task environment for the two problems varies greatly. That nature of the ultimate product of both tasks is the same, a model of a realizable physical object. The constraints on the final object differ greatly, both in content and in fundamental structure. The constraints on the design problem are quite vague. For example, in the bathroom design problem given above [Eastman 70], one constraint might be to provide privacy for the toilet. This is in great contrast to a possible constraint from the definition problem: the hole in the MBB Gehause part must be concentric with the outside curve. Constraints likely to be found in design problems are often geometrically non-specific, that is the constraint may not be specifiable in terms of the geometry of a part or assembly, but rather in some other set of terms, for example performance.[4]

Comparing problem spaces for the two problems is no mean task, as a problem space is, in final analysis, inferable only from observation of actual problem solving behavior. However, certain characteristics of problem spaces for design and for definition should be inferable from the characteristics of the example problems ,given above, and their task environments.

Taking the five components of a problem space individually and discussing each with respect to the two task domains demonstrates the degree to which the two tasks differ.

---

[4] A specific example: a building design is constrained by structural performance requirements. These demand, for their computation, knowledge of material properties and loads.

The internal symbol structures used for design must carry very different information when compared with the definition task and are therefore likely to differ in structure as will as in content If knowledge about a design can be described as being composed of objects and relations[5] then the objects of design are representations of physical objects-to-be which are constrained in location and shape by relations to other objects. Ballay [Ballay 84] has pointed out that at any time in the design process the objects in a design can be characterized, as being at a particular point along three dimensions, inclusion, coherence and precision. Inclusion is the degree to which all conditions of a design problem have been addressed in the solution. Coherence is the degree of constraint satisfaction that has occurred amongst the various parts of a design. Precision is the degree to which commitment to a particular geometric form has been made. This can be contrasted to the objects used in the definition task, which at every point in time are well-formed representations of physically realizable parts.

The relations between parts that exist during the process of design can be described along the dimensions mentioned above. As design progresses, the information in the problem space states which relate objects becomes progressively more inclusive, coherent and precise. For example, in the bathroom design problem reported by Eastman [Eastman 70], the subject included first the main fixtures and then, only when these were committed, did he place the mirror, medicine cabinet and and towel rack. As the subject progressed, his designs became more coherent, for example, the issue of privacy became progressively more resolved. Finally, concerns of fine dimensions with respect to the human figure occur at the end of the design session, rather than at the beginning. Compared to the definition task where relations between parts are precise locational relationships, the relations of design are both less complete and more changeable in both structure and content over time.

Information other than that required to physically build an object is also used in representations during the process of design. A designer may make marks which do not directly signify an addition or change to an object, but rather are referents (or icons) to internal concepts relating to the object These referents include explanatory notes, surface renderings, graphical iconic marks, grid lines and other forms of information not directly a part of a model of the object They are traditionally two dimensional but need not be so restricted when a computer medium is used. These marks, as the first commitments toward an idea that may eventually become physically realizable, are essential to the process of design.

That the operators used on objects in the design task differ from the define task can be argued on the basis of their operands. If the operands of design, the internal symbol structures of the problem space, contain constructs which differ from those of the definition task, then the operators on those constructs will also differ. In particular, these operators will be geometrically less precise for the design task as opposed to the definition task. For example, in design objects may be placed "beside" other objects without-determining a precise location for them. This is in contrast to the operations which occur in the process of definition of an object where commitment to precise location is made at every step. ~

---

[5] There is significant evidence for this conceptual division. Space allocation problems [Hemming 78] [Hemming 85] [Mitchell 77] are traditionally specified in terms of units and relations. While not conclusive support for a "natural" classification into objects and relations, the mere fact that designers of space allocation problems have universally proceeded with the assumption of objects and relations is strong circumstantial evidence. The cognitive psychology literature is on visual imagery uses concepts of "units and relations" [Kosslyn 83] to build models of visual image processing. Many diagramming techniques, for example, bubble diagrams, taught in undergraduate architecture curricula have a strong "objects and relations" flavor.

Another way in which operators differ between the two tasks is in the the information that is changed with each operation. If we look at operators as incremental changes to an object representation, then an operator can be viewed as substituting for some part of a representation a new part This substitution leaves some aspects of a representation unchanged and acts to change others. In design, as opposed to definition, the information may change in fundamentally different ways during these substitutions. For example, in Eastman's bathroom problem [Eastman 70], the subject at one point substituted two sinks in a corner vanity for a single stand-alone sink. This substitution preserved the function of the sink while changing virtually every aspect of its geometry. Compare this to making a hole in the MBB Gehause part where only local and comparatively "simple" changes to the geometry may occur.

In the third component of a problem space, the initial knowledge state, the differences between the two tasks are strongly pronounced. In design, the initial knowledge state is a statement of some conditions, which may include geometric information (a site), material information (materials of objects on the site) and a human condition (a developer owns the site). In the definition task the initial condition is complete geometric information describing an object in some other representation (i.e. -possibly drawings).

The fourth component of a problem space is the problem itself. In design, it is stated in terms that may be far removed from the geometry of an object, for example, a description of some desired performance. For the definition task the problem can be much more simply stated, that the final representation be informationally equivalent to the initial one.

The last part of a problem space is the knowledge required to effectively solve the problem. In design, this knowledge set is large; for the bathroom problem knowledge of costs, building code requirements, human ergonomics, materials and building construction is needed (and all this for a very simple design problem!). In order to complete the definition task, the required knowledge is much less and is limited to knowledge of three dimensional operations on polyhedra.

Table 1 summarizes the differences between the problem spaces required of the two problems.

It seems reasonable to posit differences in the type of search used in the design task as compared to the define task. The define task should be characterized by problem solving closely akin to the "programmed activity" referred to in Newell and Simon [Newell 72,p.822]. This type of activity can be described as the use of a highly specific algorithm in the solution of a problem, particularly an algorithm which involves little search and back-tracking. An example of such problem solving is given by Hillyard [Hillyard 82] where the intermediate states required to created the MBB Gehause part are shown in Figure 2. The transformation required to create each state form the previous one is clear, and in general tfie'effects of a particular operator can be predicted in advance, making the path to a problem solution straightforward. This can be contrasted to the task of designing a bathroom as reported by Eastman [Eastman 70] in which thirteen significant branches in a problem behavior graph occurred in the course of a twenty five minute protocol. These differences in both the amount and type of search display emphatically the differences between the two tasks.

Finally, different types of external memory would be useful to the two different tasks. Each task requires an external memory which can effectively capture the type of information used in the problem space of the task and which supports the type of search activity used. As we saw earlier, these differences are extensive. Table 2 summarizes the differences between the two types of problems.

PROBLEM SPACE COMPONENT	DESIGN	DEFINE
A set of symbol structures each representing a knowledge state	Information needed are incomplete models of geometric objects	Information needed are complete, well-formed polyhedral representations
	All information may not relate directly to a buildable part	All information directly relates to physical parts
A set of operators to incrementally generate new knowledge states	Operators act upon incomplete information and create new incomplete information	Operators work only on complete specification of geometry
	Operators may make large geometric changes	Operators make incremental geometric changes
	Operators apply to non-spatial representations also	Operators apply to spatial representations
An initial knowledge state	Statement of some conditions which may not be spatial	Complete spatial information in some other representation
A problem, (specification of a desired knowledge state)	Test for solution stated in ill-defined terms which may not be geometric	Test for solution very well defined
A set of knowledge	Needed knowledge set is large and not given in problem statement	Needed knowledge set is strictly geometric

Table 1: Comparison of Problem Spaces for the Design and Define Tasks

STRATEGIES FOR SUPPORTING DESIGN

Consider that design can be represented using the theory of human problem solving as referred to above. That theory would represent design as a model consisting of a human information processing system searching in a problem space which is largely determined by a task environment The behavior of the overall model is affected by its structure, its overall organization and its parameters (for example, the capacity of short term memory (STM)) The value of that behavior, or the *performance* of the model, can be measured by:

THEORY COMPONENT	DESIGN	DEFINE
The human information processing system	Invariant	Invariant
Search	Search is branching and backtracking	Programmed search
Problem space see table 1	Complex, large Ill-structured	May be complex and large Well defined
Task environment	Constraints and requirements are vague	Constraints are well-defined
External memory	Reflects differences in search, problem space and task environment	Reflects differences in search, problem space and task environment

Table 2: Comparison of the Design and Define Tasks

- Its *efficiency* in both time and space. Efficiency is concerned with the resources that are used to arrive at a problem solution, not the value of the solution itself. A measure of the efficiency is the amount of time required to arrive at a solution to a given problem.

- Its *power,* or ability to find good solutions to problems. Measures of the power of a process would be taken on the usefulness of the product, the designed artifact, as a solution to the problem.

An increase in either performance characteristic of the model without a decrease in the other would be an unequivocal improvement in performance. Strategies for supporting design using interactive tools should therefore focus on improving either the efficiency or the power of the design process (or both).

These strategies may be implemented by using a computer program to change either the parameters or the structure of thetnodel. This is exactly the purpose of a computer support to a process: that the computer should perform some task or aid in its performance to increase the usefulness of the system as a whole. An example of a change to the parameters of the model would be a computer system that could increase the speed of access of external memory. If that speed were increased to a rate comparable to the access time for STM, then the capacity of STM, a limiting parameter in the behavior of the system, would effectively be increased. An example of a change to the structure of the model would be the adoption of a stronger method than heuristic search for generating solutions, This would change the fundamental nature of the problem solving process so that it might no longer be described as search in a problem space. Various techniques of mathematical programming, for example, linear programming, fall into this category.

Taken together, strategics for increasing cither the power or the efficiency of the human designer and techniques for implementing those strategics which change either the parameters or the structure of the model, suggest a classification, in the form of a two by two array, of computer assists to design. Figure 4 diagrams the framework for such a classification.

	STRATEGY FOR:	
METHOD MAKES:	increasing efficiency	increasing power
change to parameters		
change to structure		

Figure 4: Classification of computer assists

In the following paragraphs are described four specific approaches to computer assists. In each case the main contribution of the approach with respect to the classification above is identified and the approach is described at length.

Search Strategies

Search is an iterative activity that requires four decisions to be made at each iteration. Informally these are:

1. Is the problem solved? If it is, stop and declare a solution, else continue.

2. Which state of knowledge will be transformed next? If no states of knowledge can be transformed, stop and declare that no solution can be found, else continue.

3. Which operator will be used on the selected state? Apply the operator.

4. Which of the existing states shall be remembered for future reference? Remember the chosen ones.

A search strategy consists of a set of policies with regard to each of these decisions. A computer can assist in search by some combination of assisting in making decisions and taking autonomous responsibility for decisions. Thus a computer assisted search strategy is one in which the policy for one or more types of decision has a computerized component. An examination of each decision type under either a computer assisted or a completely computerized decision policy yields eight simple options for computer assisted search.

1. Computer assisted determination of a problem solution. An approach to this option would provide the computer with a number of evaluation algorithms which could be called by the designer to examine a proposed solution to see if it meets solution criteria. Another approach would compute and describe the relevant tradeoffs of one proposed solution

with respect to other already generated proposed solutions[6].

2. Computerized determination of a problem solution. Having a computer make all tests of adequacy of a solution will work when the solution tests have some strong properties:

   - All solution criteria are well-defined and amenable to computer analysis, and
   - Either:
     - o All solution criteria can be simultaneously met in a solution (strict satisficing).
   - On
     - o All solution criteria can be combined into a single measure of value. This measure of value can be unambiguously compared against the same measure computed for other possible solutions.

3. Computer assisted selection of a knowledge state. Selection of a knowledge state can be predetermined by the decision policy in the search strategy, or it can determined by some knowledge of a problem[7]. A computer assist would be more helpful in the second case. One approach would be a program to display the search tree thus far generated so that a designer was better informed of the opportunities and alternatives available. Another approach would be to evaluate each of the states with an *evaluation function* which would produce a measure of the likelihood of each state being a solution precursor.

4. Computerized selection of a knowledge state. Selection of which knowledge state to select can be entirely automated if either:

   - A weak method of search is used, in which case the selection is trivial, or
   - A strong and reliable test for a solution precursor exists.

5. Computer assisted operator selection and application. An approach to this option would be a display of all applicable operations with some prediction of their impact on a knowledge state. The designer would select an operation and the computer would apply it to the knowledge state. A weaker form of assistance could be provided by the computer displaying the operations that have already been applied to a knowledge state. The designer would use that information to formulate and apply the operation manually.

6. Computerized opejator selection. One approach to this option is the operator selection mechanism of means/ends analysis, in which operator selection is mapped to selection from a *table of differences.* In this table an operator is selected if it will best reduce an observed distance between a current state and a goal state. Another automated approach to operator selection is that of a knowledge driven system. Whenever the knowledge state to be operated on exhibits certain characteristics an operator may be specified. This approach lacks any formal explanation, as the characteristics which cause operator

---

[6] HTiis is the approach of Pareto optimality.

[7] This is exactly the distinction between the weak methods of search and strong knowledge based methods.

selection can be completely arbitrary. This fact has not stopped application of the idea in many knowledge based systems based upon *rules.*

7. **Computer assisted selection of knowledge states to retain.** A key idea for this option is the notion of access. As a designer develops a series of knowledge states, the computer could provide a means to characterize these states in some manner. This could be in the form of a naming convention, or could be automatic, where the state is characterized and indexed by algorithms. Version control ideas from the database field will likely be applicable here.

8. **Computerized selection of knowledge states to retain.** When weak methods of search are used, their exist simple policies for determining which states to save.

Each of these strategies for supporting search could be combined with other strategies to form more complex models of computer-assisted search. All options share the characteristic of moving some of the main cognitive activity of design, that of search, into the realm of a computing device. Thus all of the options change, to varying degrees, the structure of the model of problem solving. Some of the options, particularly those of selection of knowledge state and operators, have the effect of making feasible explorations which would be intractable by manual methods. These options act to increase the power of the designer.

Similarity Between External and Internal Representations

When designers solve problems, they use an internal formulation of the problem, known as a problem space, to carry out the information processing required to solve the problem. Almost always, designers make use of an external memory, in the form of drawings or models, to assist their memory and to help structure the process of exploration in design[8]. The use of this external memory implies a continual translation between it and the representations internal to a designer's mind. It can be shown [Card 83] that the cost, in terms of time spent translating, can increase dramatically as the difference between the internal and external representations grows. The strategy then, for creating design systems, is to discover and provide external memory representations that are directly mappable into internal problem space representations.

These representations are likely to take the form of *abstractions.* An *abstraction* is a partial representation of an object, that expresses some information about the object and suppresses all other information. A well-known abstraction in architectural design is the bubble diagram, that expresses adjacencies (and possibly size) without strong specification of geometric shape and location[9]. Figure5 shows a plan representation of a simple building and its associated bubble diagram. Abstractions such as these speak more directly to a designer than do complete descriptions, where abstract information of this type can be easily lost.

External representations of internal concepts are more than just a means to record information in a succinct and convenient manner. Once so recorded in the external world, they can be subjected to a whole range of operations and queries that are not possible within the limited scope of a designer's unaided cognition. They can be changed and manipulated according to explicit rules; their properties

---

[8] This external memory is necessary because of inherent limits in the human information processing system.

[9] Unfortunately, bubble diagrams also convey some, usually unintended, information about the placement and shape of objects. It is an all too common failing of novice designers to merely "round off" the corners of a bubble diagram and call it a "solution".

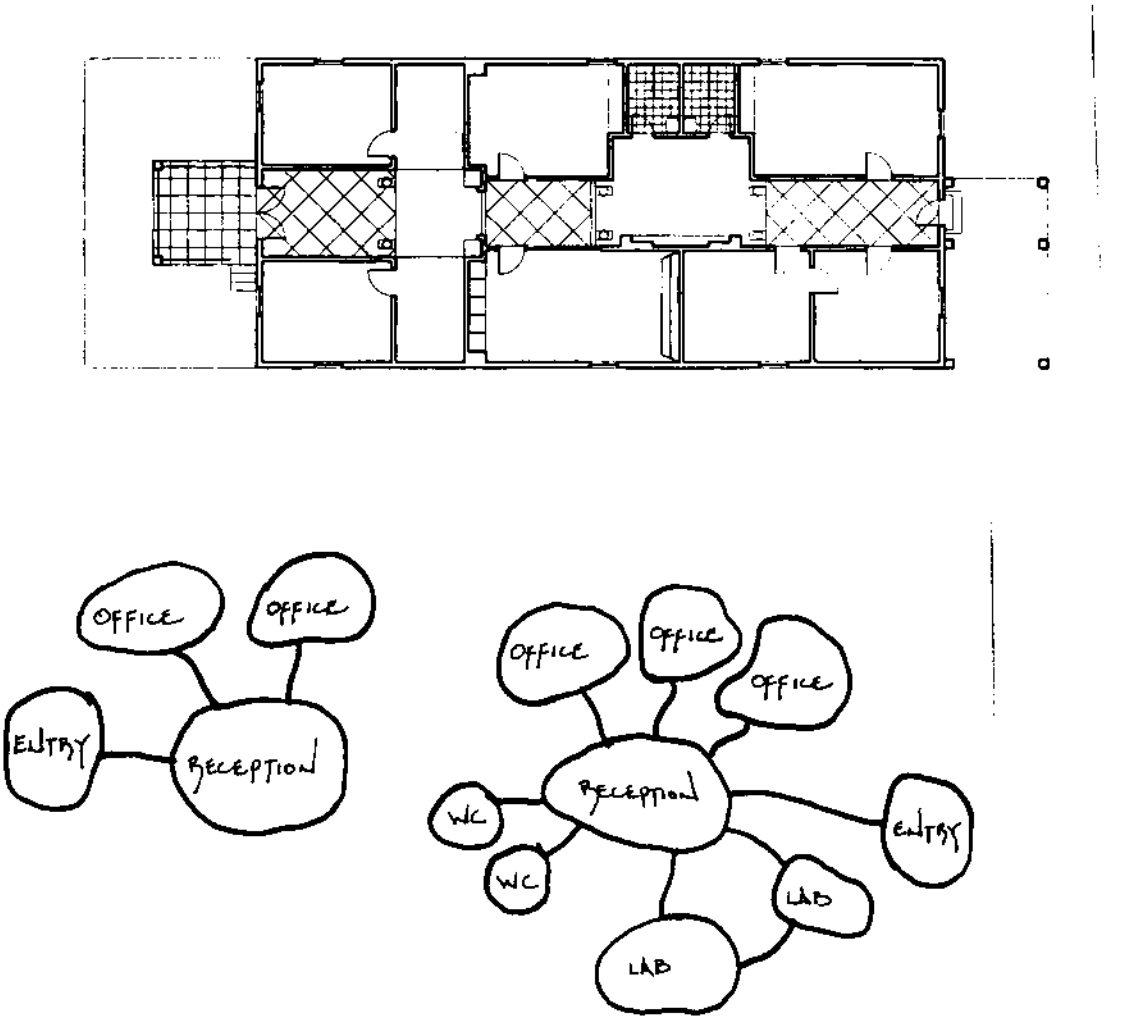


Figure 5: A floor plan and its associated bubble diagram

and implications can be formally examined. These external operations can greatly increase the utility of a representation. If a representation is both a good model of the problem space in a designer's mind and can be given formal interpretation as a mathematical object, then a great opportunity emerges. The representation can be systematically examined and used to compute its inherent implications. An example of such a symbiosis is found in the work of Flemming [Flemming 85] who uses rectangles and a simple relation set as a representation of diverse design situations. One priority for implementation of this strategy is the discovery and formalization of a useful set of abstractions.

Designers tend to use many different, often idiosyncratic, abstractions as they design. Not all of these abstractions can be foreseen by a system designer, nor are all of these representations formalizable. As an example of the richness of different abstraction types see [Laseau 80,chapter 4]. An approach to the first of these problems, the lack of a complete representation set is the combination of existing relations to build new relations. A first example is [Akiner 85]. The second of these problems seems less tractable. The benefits of a formalizable representation are the external computations that can be performed on it. If a representation is not formalizable then the remaining benefits are simply as a repository of knowledge in external memory. This benefit could possible be achieved by a graphic "language" convention as described by Laseau [Laseau 80,chapter 4].

The strategy of making external representations that are close analogues to internal problem space representations falls into two categories in the classification of Figure 4. By making the external mirror the internal representations, access time to external memory is reduced, thus increasing the efficiency of the designer. By creating formal external models which can be computed, the structure of the model is changed, to a structure in which the external memory is no longer passive. This change can have an effect on the power of a designer by allowing otherwise infeasible operations to occur.

Parsimonious Input

An external medium is used in manual methods of design to augment memory. A designer interacts with that medium by creating marks on paper or by performing operations on a physical model. In some situations, very brief informal marks may be laden with meaning, in other situations, for example perspective rendering, a great deal of effort may be placed into the creation of a single representation. The early stages of design tend to make use primarily of marks of the first variety, i.e. marks that with great parsimony of effort capture their intended meaning. Just as a designer seeks to use his manual mark making parsimonious or efficient, an interactive computer system should provide very parsimonious means of creating its representations.

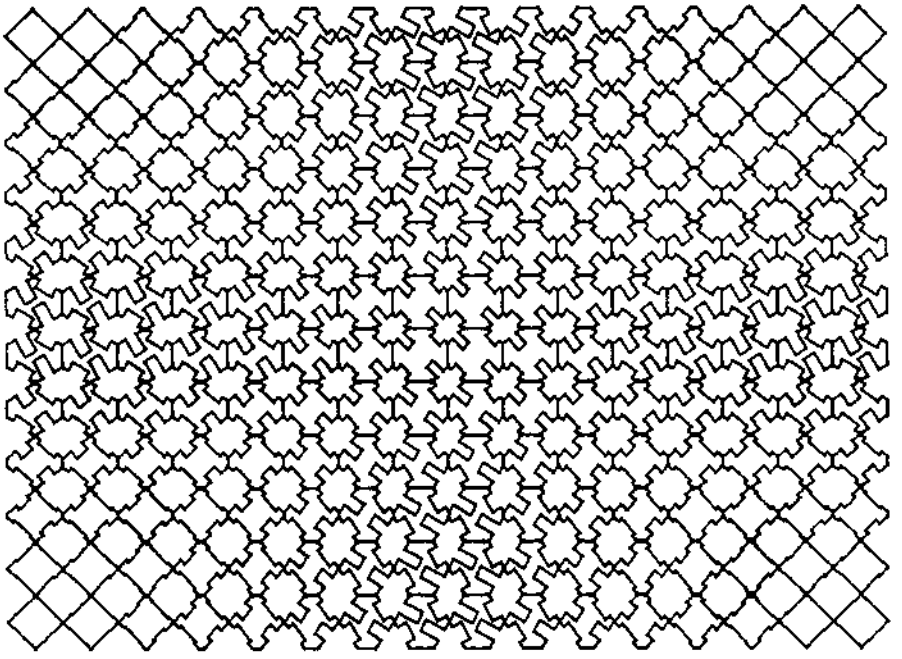


Figure 6: A tiling of the plane

One of the greatest sources of parsimony of input in design could be the capture of repetitive types of representations. For example, in designing a window facade, a designer using manual media must draw each facade element (that he chooses to represent) individually. Yet these elements are highly similar and the intention of the drawing activity can be captured in a single phrase, "put windows in the facade". Another example is in the creation of tilings of the plane, a design exercise originated by William Huff [Hofstadter 83] and recently used in a different form in a course, developed by the author, entitled *Computer Modelling in Design* at Carnegie-Mellon University. These tilings, a computer generated example of which is shown in Figure 6, are the product of exploration of

sequential transformations on an initial tiling. At each stage in the exploration, a representation in the form of a drawing is made so that the effect of the latest transformation can be assessed. When done manually, these drawings consume a great deal of time and effort An ability to describe the transformation directly, in terms of an operation on some part of an existing tiling, would reduce the effort required for drawing, allowing a designer's attention to be focused more upon notions of transformation.

Computer programming languages provide constructs for expressing the regularities in form and transformation described above. They are a model for formally representing sequence. Many geometric modelling systems have been based on computer programming paradigms [Fitzgerald 81] [Brown 82] yet none, to the knowledge of the author, has brought the process part of the programming paradigm into the interactive graphical interface in a strong way. A system which supports interactive design should have explicit mechanisms for expressing and executing algorithms. These mechanisms should be presented graphically to the user.

With respect to figure 4 these strategy seeks to improve the efficiency of the system by changing the parameter of speed of writing on external memory.

Design as Note Taking

The representations of design can be viewed as *notes,* which singly give some information about a design and in total specify a design to some level of completion. The term *notes* is used to emphasize that most design representations are informal, that is they may not follow consistent conventions and they may not be complete in the event that a convention is consistently used. In addition they may contain information extraneous to the specification of a design, but essential to the process of creating the design. This informality serves important purposes for designers:

- It allows the recording of ideas without a large time and effort commitment
- Levels of informality can be used as a "code" to indicate the degree of commitment which has been brought to an idea. A drawing at small scale, done quickly, with thick charcoal is easy to change or abandon compared to a complete plan drawing, with dimensions done with ink on mylar.
- It allows for rapid change of information structuring, format and style; all important for a designer using the type of search described under the Newell and Simon model.

The computer support of note taking is problematic. In order for a computer supported note to be more useful than its manual counterpart, some of the information conveyed by the note must be extracted and explicitly* represented. Yet a fundamental property of notes is their informality, their seeming non-adherence to a well formed syntax of representation. One very speculative approach to this problem is an "opportunistic" note recognizer. A system of this sort would operate by recognizing patterns in a note and making inferences from those patterns. The validity of these inferences would be manually checked. In order to work, such a system would require a strong pattern recognition facility and an ability to represent complex rules about patterns. The production system paradigm fits these requirements, but current production systems do not operate on the bitmap kind of representations that would most likely be used as the medium of note communication.

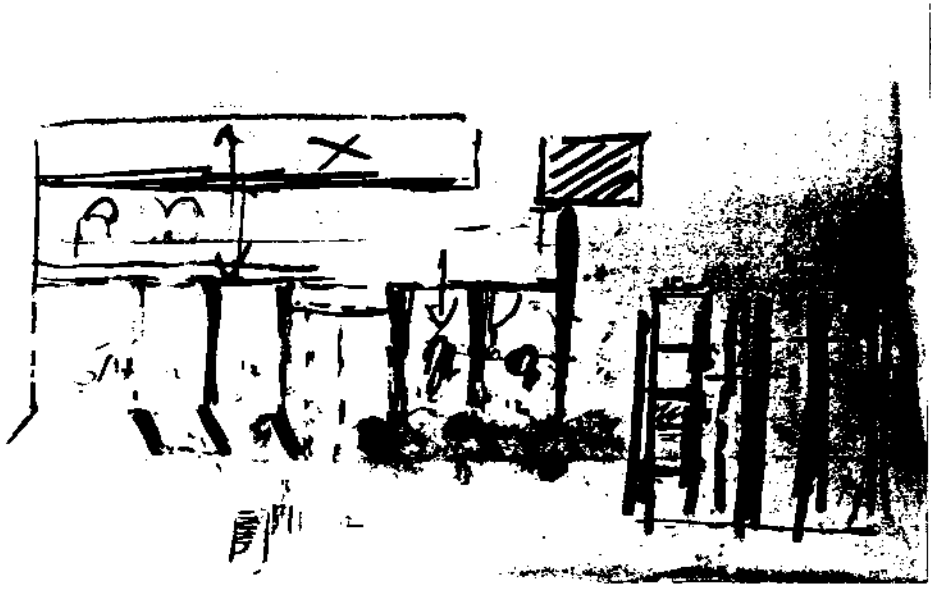


Figure 7: A preliminary sketch of the Salk Institute. *LKahn*

Figure 8: The **Salk** Institute **as built**

With respect to the classification in figure 4 this approach seeks to change the efficiency of the **human** information processing system (IPS) by changing the structure of the external memory that aids **the** designer.

STRATEGY FOR DEVELOPMENT

Creating modelling systems to support the interactive generation of designs is a long range task. Priorities for inquiry in this area need to be established if such systems are to be created.

When computer tools are newly introduced to a field, increased efficiency of existing methods has traditionally been the first goal of their use [Akin 84]. Eventually though, computation has a more profound impact on a field, for it begins to change both the structure of the methods used and the nature of the end products. A prime example is civil engineering, where computerized finite element methods introduced for the first time, a strong notion of iterative design, and the ability to "fine tune" or optimize structural performance. Similarly, in looking for strategies for developing interactive design systems, one should look first to those ideas which hold the most promise for restructuring method and changing product and secondarily to those ideas which merely seek to increase the efficiency of design.[10] This suggests that those strategies aimed at increasing the power of the human IPS (the rightmost column of figure 4) show the most promise for developing systems for interactive design. Completing Figure 4 as Figure 9 shows that two of the strategies reported here, *search* and *internal/external* have an influence on the power of the designer. They both act by removing some of the cognitive burden from the designer and placing it in external memory.

	STRATEGY FOR:	
METHOD MAKES:	increasing efficiency	increasing power
change to parameters	internal/external parsimonious input	
change to structure	note taking	internal/external search

Figure 9: The strategies classified

Both of these strategies are worthy of pursuit. Search seems to be the most likely candidate for first exploration as there exists, in the field of artificial intelligence, a strong framework in which search can be studied. Thejuse of search augmenting methods in an interactive design system relies on explicit representation of the domain being searched. This representation must have certain formal properties dependent upon the type of decision that the computer system is making or assisting. Today there appear to exist only two representation paradigms which are sufficiently formalized to fill this role; rectangular layouts [Flemming 85] and shape grammars [Stiny 80]. More research into formalized representations seems to be in order.

---

[10] This statement is not meant to denigrate important and essential inquiry into highly interactive operations. Rather it argues that the idea of a truly interactive design system is young and as such the limits of the idea are not fully explored. The most profound effects are liable to arise from changes to methods and products rather that efficiency of process.

If these new formal representations are to be truly useful in design, they should map easily to internal representations that designers can maintain. Thus the strategy for making external mirror internal representations should be given a high priority. Two sources for insight into such representations seem available. One is the research into the ways in which designers work with manual methods [Akin 79]. The other would lie in investigations into representations designers could use were they to have suitable supporting external representations. This second approach is more speculative, but is closer to the goal, of creating new synergies between person and computer. Making external and internal representations similar can also have the effect of transferring some of the cognitive burden away from the designer onto the machine. However, there does not exist a firm intellectual framework within which to understand how these activities would affect design, so first efforts in this area are likely to be *ad hoc.* A worthy goal would be the creation of such a thought structure that would explain in detail the role such external representations can play. These strategies, for increasing the power of the designer to create solutions, seem the most promising avenues of inquiry into the support of interactive design systems.

The other two strategies for interactive design systems fall into the categories, shown in figure 9, for improving the efficiency of the design process. This classification provides no means for ordering their importance, so no such ordering is reported here. Some connections to other fields should, however, be noted The strategies, for parsimony and note-taking seem related One goal for creating notes is to achieve a low time commitment for their construction, exactly the same goal that parsimony of input pursues. Ideas of iteration from programming languages seem appropriate here. These ideas have been formulated in a domain other than geometry. Investigation of programming ideas in a geometric domain could provide significant insight for programming language research. One approach to accomplish the note taking strategy would be a bit-map pattern recognizer. Successful development of this idea would provide insights for the field of pattern recognition in general and computer vision in particular.

In summary then, strategies for the development of systems for design are dominated by those that promise to change both the structure and the product of design. These include those strategies aimed at search and at mapping between internal and external representations. Other strategies, for increasing efficiency of design, seem to have no rank ordering of importance yet are interrelated

The four strategies discussed here certainly do not exhaust the possibilities for approaches to supporting interactive design systems. Other ideas for building such systems can be imagined; many are latent in the extant literature and in the bewildering variety of interactive computer programs which exist today. What seems to be lacking is a conceptual structure which can give form to research efforts. A *theory* is needed; a theory which takes as its starting point the goal of creating interactive design systems, and explains in detail requirements, organizations and means of evaluation. Many of the pieces of suchra theory exist today. Current understanding of designer behaviour reinforces the central notions of the Newell and Simon theory and extends those notions into more complex types of problem solving. Studies in human computer interaction provide a comprehension of the detailed, low level interaction that occurs between humans and machines. Research in automated design provides an inventory of models which suggest possible system designs. If interactive systems which truly support design are to be built, these disparate pieces must be pulled together and knit into a coherent whole.

References

[Akin 79] Akin, Omer. *Models of architectural knowledge: An information processing model of design.* PhD thesis, Carnegie-Mellon University, 1979.

[Akin 83] Akin, O., Baykan, C., Rao, R. Searching in the UNIX Directory Space. 1983.

[Akin 84] Akin, O., Flemming, U., Woodbury, R., Development of Computer Systems for Use in Architectural Education. 1984.

[Akiner 85] Akiner, V.T. Topology - 1: A Reasoning System for Objects and Space Modelling Via Knowledge Engineering. In *Design Engineering Division Comference on Mechanical Vibration and Noise.* American Society of Mechanical Engineers, Cincinnati, Ohio, September, 1985.

[Ballay 84] Ballay, J.M., Graham, K., Hayes, J.R., Fallside, D. *Final Report Phase IV : CMU/IBM Usability Study.* Technical Report, Carnegie Mellon University, March, 1984.

[Brown 82] Brown, Christopher, M. PADL-2: A Technical Summary. *IEEE - Computer Graphics and Applications* 2(2):69-83, March , 1982.

[Card 83] Card, S.K., Moran, T.P., Newell, A. *The Psychology of Human-Computer Interaction.* Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1983.

[Eastman 70] Eastman, Charles M. On the Analysis of Intuitive Design Processes. *Emerging Methods in Environmental Design and Planning.* MIT Press, Cambridge, Mass., 1970, pages 21-37, Chapter 3.

[Fitzgerald 81] Fitzgerald, W., Gracer, F., Wolfe, R. GRIN: Interactive Graphics for Modelling Solids. *IBM Journal of Research and Development* 25(4):281-294, July, 1981.

[Flemming 78] Flemming, U. Wall representations of rectangular dissections and their use in automated space allocation. *Environment and Planning B* 5:215-232, 1978.

[Flemming 85] Flemming, Ulrich. On the representation and generation of loosely-packed arrangements of rectangles. *Planning and Design* , December, 1985.

[Galle 81] Galle, Per. An algorithm for exhaustive generation of building floor plans. *Communications of the ACM* 24:813-825, 1981.

[Hillyard 82] Hillyard, R.C. The Build Group of Solid Modellers. *IEEE - Computer Graphics and Applications* 2(2):43-52, March, 1982.

[Hofstadter 83] Hofstadter, Douglas R. Metamagical Themas. *Scientific American* 249(1):14-20, July, 1983.

[Kosslyn 83] –Kosslyn, Stephen M., Reiser, Brian J., Farah, Martha J., Fliegel, Sharon L. Generating Visual Images: Units and Relations. *Journal of Experimental Psychology: General* 112(2):278-303, February, 1983.

[Laseau 80] Laseau, Paul. *Graphic Thinking for Architects and Designers.* Van Nostrand Reinhold Company, New York, N.Y., 1980.

[Mitchell 77] Mitchell, William J. *Computer-Aided Architectural Design.* Petrocelli/Charter, New-York, 1977.

[Newell 72] Newell, A., Simon, H.A. *Human Problem Solving.* Prentice-Hall Inc., Englewood Cliffs, N.J., 1972.

[Reitman 64] Reitman, W.R. *Human Judgements and Optimality - Heuristic decision procedures, open constraints and the structure of ill-defined problems.* Wiley, New York, 1964, pages 283-315, Chapter 15.

[Simon 73] Simon, H.A. The structure of ill-structured problems. *Artificial Intelligence* 4:181-201,1973.

[Stiny 80] Stiny, George. Introduction to shape and shape grammars. *Environment and Planning B* 7(3):343-352,1980.

[Sutherland 63] Sutherland, I.E. *Sketchpad: A Man-Machine Graphical Communication System.* Technical Report 296, MIT Lincoln Lab., 1963.

[Woodbury 85] Woodbury, Robert F. Computer Techniques for Designers: Strategies. In *4th Canadian Building Congress.* National Research Council, Ottawa, Ontario, October, 1985.

List of Figures

List of Tables