GENERATIVE DESIGN IN ARCHITECTURE USING AN EXPERT SYSTEM

Eric Gullichsen
Ernest Chang
Dept. of Computer Science,
P.O. Box 1700,
University of Victoria,
Victoria, B.C. V8W 2Y2

March 1, 1985

ABSTRACT

The mathematician-architect Christopher Alexander has devised a theory of objective architectural design. He believes that all architectural forms can be described as interacting patterns, all possible relationships of which are governed by generative rules. These form a 'pattern language' capable of generating forms appropriate for a given environmental context.

The complexity of interaction among these rules leads to difficulties in their representation by conventional methods. This paper presents a Prolog-based expert system which implements Alexander's design methodology to produce perspective views of partially and fully differentiated 3-dimensional architectural forms.

RÉSUMÉ

Le mathématicien-architecte Christopher Alexander a inventé une théorie d'esthétique architectural. Il croit qu'on peut écrire toutes les conformations avec un systéme de régles qui agit l'un sur l'autre. Les régles se forment un language qui peut produire les conformations convenable au contexte de l'environnement.

C'est difficile de représenter ces régeles par les méthodes classiques à cause de la complexité d'interaction entre les régles. Nous presentons un systéme qui utilize la langage de programmation Prolog pour mettre en oeuvre la méthodologie d'Alexander et créer les perspectives tridimensionelles des conformations partialement et totalment distinguées.

KEYWORDS: architectural design, expert systems, logic programming, generative systems.

'From the sequence of these individual patterns, whole buildings with the character of nature will form themselves within your thoughts, as easily as sentences.' [3,p.xiv]

1. Introduction: The Pattern Language of Christopher Alexander

1.1. Patterns: Forms within a Context of Forces

What are the primitive 'atomic' elements which comprise an architectural form such as a room, a building, or a city? In answering this question, the environmental context of what may upon first consideration appear to be atomic elements must not be neglected.

According to Alexander, patterns, which consist of static physical elements (the form) in the environment, together with dynamic occurences related to the physical elements (the context) which in turn result from the interaction of certain relevant forces, constitute the entire physical substance of the world. To consider the world to be made of physical 'things' entails a misconception analogous to that held by classical physics in considering an 'atom' a thing. Western languages contribute to the perpetuation of the illusion that 'things' are fundamental, by their preponderance of nouns. In Alexander's view, a noun is simply a convenient label for a set of relationships amongst patterns.

What is it then that is repeated in an architectural form? Traditionally, physical entities are repeated hierarchically: an apartment building can be seen as a collection of suites, each of which is a collection of rooms. However, this classical view fails to explain how or why these elements tend to be associated with different sets of events. For instance, people who live in an apartment building do disparate things in their respective suites, and the modular viewpoint fails to account for this variation.

Alexander asserts that hierarchies of relationships, rather than of things, contain the architectural primitives we seek. Each pattern is a morphological law which defines a permissible set of spatial relationships within a given context. The taxonomy of these morphological laws is a hierarchy, as each pattern is itself a pattern of relationships among other patterns.

The relationships between patterns are the only entities repeated in the world. Patterns, consisting of a space and the events which occur there, are the atoms of the man-made universe. According to Alexander [3,p.98] these patterns are only a few in number: a building is defined by several dozen patterns.

If this is accepted, the obvious question is then how these canonical patterns combine to produce the world we know. Before we attempt to answer this, a distinction must be made between 'good' and 'bad' patterns.

1.2. The Quality of Patterns

Alexander claims that patterns vary from 'good' to 'bad', and that the quality of a form to be created can be assured by objective scientific means. Good patterns possess a certain hard-to-characterize quality of holistic completeness or 'life' which Alexander terms the 'quality without a name'. Patterns which produce irreconcilable conflicts in humans and thereby increase their psychological stress are bad patterns. What then is a good pattern? As will become apparent, good patterns are those generated by a specific set of rules which take into consideration the forces acting on the pattern being designed.

Such a pattern has an ecological balance of internal and external forces acting upon it. Where this balance is stable, as in the forms found in nature, a pattern is good or 'alive'. These live patterns can interact to support each other.

Natural objects are always formed by the forces which arise within them. Objects created by man are also formed through the action of certain forces, but there may also exist additional latent forces which do not directly influence the form of the object. However, a design procedure which does not account for all forces acting on objects will inevitably lead to an unstable system.

Living patterns are easily recognized from their geometric character. Alexander stresses that a hierarchy of living patterns is never modular. Rather, the reconciliation of patterns with their internal forces make their details unique, like the leaves of a tree, or the waves of an ocean. To be whole and alive, buildings and other architectural structures must have this natural characteristic of responsiveness to internal and external environmental forces.

It is precisely this holistic characteristic of good architectural design requiring the interaction of large amounts of knowledge that makes the use of expert-system techniques suitable.

1.3. The Generation of Pattern Hierarchies by Pattern Languages

Good patterns cannot be brought into existence by a single monumental effort of intellect, but only through process. Just as life within a natural organism implies the maintenance of a balance of forces, so a building which is alive must be grown from a set of patterns in which the parts created are harmonious both internally and in their totality. Rules for combining patterns constitute

> 'a way of focusing attention on some particular holistic behavior in a thing, which can only be understood as a product of interaction among the parts.' [1]

Alexander claims to have discovered a simple set of generative rules which determine the structure of any environment. These rules are similar to a genetic code, and govern human acts of building. Architectural forms generated by this 'pattern language' are necessarily 'alive' since a balance of relevant internal forces follows inevitably from the manner of their creation. Details of the 253 patterns which comprise this pattern language have been published [2]. Alexander asserts that the language represented by these patterns constitutes the archetypal core of all possible pattern languages.

He makes a clear distinction [7] between a generative design process, and design governed by constraints, which is the conventional method of formulating design rules in architecture. An appropriate analogy can be found in the field of linguistics. Chomsky [4] was the first to develop generative grammars for languages, both natural and artificial. Until Chomsky, most grammatical rules were expressed in the form of constraints which sentences in the language obeyed. Chomsky's generative specification of grammatical rules was novel in that only correct sentences would ever be created. Therefore, complex and uncertain 'generate-and-test' methods were no longer required. A similar process occurs in nature, in which an organism grows from an embryo in accordance with the generative rules encoded in its chromosomes. The same efficient design methodology is possible, according to Alexander, in architecture.

2. From Forces to Forms: Design with a Pattern Language

We have postulated the need to achieve during its design process, a balance of forces acting on a designed object, for it to be 'alive' and have the 'quality without a name'. We now explore more thoroughly how this need can be satisfied through use of a pattern-based design process.

2.1. A Formalism for Patterns

In [3,p.247] Alexander gives a rigorous treatment of patterns, the basic entities of his design system. A pattern both corresponds to a certain class of thing which exists in the world, and is a rule describing the design (generation) of that thing.

The structure of a pattern language follows from the fact that individual patterns are not isolated. Each pattern occupies a position in a (possibly cyclic) network of related patterns, connected to the smaller patterns it contains, as well as the larger patterns in which it is contained. A pattern helps to complete the patterns above it in the network, and is itself completed by the smaller patterns below it.

A pattern has three components:

(1) Context. Where or when is the pattern applicable? The context of a pattern may be considered to be a set of preconditions which specify its applicability.

(2) The system of forces which define the problem solved by this pattern. Why is this pattern required? Recall that good patterns resolve or balance the internal and external forces acting on the thing designed. This component of the pattern provides the reason(s) for its application.

(3) The solution, or spatial configuration of entities implied by the pattern which permit the resolution of (2). What specifically is the invariant property common to all such solutions? As patterns are hierarchically arranged, this third component of a pattern may be highly complex, and usually involves other patterns.

Each pattern thus contains two logical statements, which must be empirically true. The first is that a given problem (2) exists within the stated context (1). Secondly, the pattern asserts that (3) solves (2). A pattern is objectively good if the problem (2) is real and configuration (3) solves (2).

To implement a design system based upon an interacting set of such patterns, the rigorous specification of (3) frequently proves to be difficult[1]. As seen in the next section, for patterns to be practically applied to a design problem, the solution may have to be a set of procedures which further differentiates the form being designed.

In [2], 253 patterns which apply to architectural forms of varying scales are presented. The scales distinguished by Alexander are those of: towns, buildings, and construction. Our expert design system employs a subset of 84 of these 253 patterns, selected to consider the design of forms from the second of the three scales, those that pertain to individual buildings and the spaces between buildings.

2.2. Use of a Pattern Language: Differentiation

The use of a language of patterns to design an architectural form involves a process of differentiation: the creation of distinctions where no distinctions previously existed. A process of differentiation which results in the 'growth' of a design should not simply consist of the addition of modular components in a hierarchical manner; each part must be modified by its position in the whole design.

Alexander's patterns are arranged roughly in order of decreasing morphological importance to ensure that a whole, imprecisely-specified form is successively differentiated during the process of design. This successive differentiation ensures that subsequent design decisions do not conflict with earlier decisions, and eliminates the need for backtracking.

Since patterns are ordered, no pattern can unexpectedly arise to act as a constraint on a partially-completed design. Patterns are applied successively in a generative manner. At each step, certain general configurations of the form are established, and details are then elaborated, conforming to the structure laid down.

More precisely, to design in a generative manner, the order of application of patterns should meet the following three heuristic criteria, listed in order of decreasing importance [3,p.380]

(1) If pattern A is above pattern B in the network of patterns, then A should be employed before B. For instance, if pattern A involves a living room, and B involves alcoves in a room, the living room design must be produced (roughly) before alcoves can be incorporated.

[1]The reader is challenged to provide a rigorous definition of a 'rough circle', an object which any child could sketch without hesitation.

(2) Before employing pattern A, all the patterns immediately above A in the pattern network should be considered in the design, as contiguously as is possible.

(3) Similarly, after employing pattern A in a design, the patterns immediately below A in the network should be considered, as contiguously as is possible.

It is the burden of the designer of a pattern language to correctly structure the network of patterns so that features which are 'dominant' in a form which can be produced are characterized by patterns which occur higher in the network of patterns[2]. As Alexander has made explicit the structure of the network of patterns for his pattern language [2], we need not concern ourselves with the problem of discovering the pattern hierarchy.

2.3. Differentiation of the Structure of an Individual Building

In order to illustrate more clearly what is meant by the process of design by differentiation, let us consider in detail that subset of patterns (104-204) which deals with the design of individual buildings.

An examination of these patterns as presented in [2] reveals that the use of this portion of the pattern language involves 8 identifiable steps of differentiation of detail of the form being designed.

(1) Initially, the position and rough shape of building(s) on the site is fixed. (patterns 104-109).

(2) Entrances, gardens, courtyards, terraces and roofs are laid out. (patterns 110-118).

(3) The gradients of space within the building are established. (patterns 127-135).

(4) Within building wings, the most important areas and rooms are defined. (patterns 136-145).

(5) The inside of the building is knit to the outside, by treating the building edge as a distinct place. (patterns 157-168).

(6) Minor rooms and alcoves are attached, to complete the main rooms. (patterns 179-189).

(7) The size and shape of rooms and alcoves are fine-tuned, to make them precise and constructable. (patterns 190-196).

(8) Finally, the walls are given depth as necessary for alcoves and windows. (patterns 197-204).

[2]It is reasonable to imagine that if patterns were represented by means of a sufficiently uniform and rich description language, an inference of the structure of the pattern network might proceed automatically.

For the design of any form which is to be a complete building, patterns are usually selected from each of the 8 groups outlined above. Variation in the exact selection of patterns leads to variation between individual designs.

3. The Prolog-based Expert System

The ideas and structure of Alexander's system of design as based on a generative pattern language have been presented. The entities of the system are the patterns, each of which has three principal constituents (section 2.1). A network of patterns is seen to form a language for design, where a useful language should be both morphologically and functionally complete. The structure of the network is governed by the morphological dependencies present between patterns. Heuristic design rules for traversing the network to apply patterns one-at-a-time in a top-down differentiating fashion have been discussed.

Computer-based experiments with generative architectural design were conducted using the very high-level logic programming language, Prolog [6,8]. CProlog Version 1.4 was employed on a VAX 11/750, and a Raster Technology Model 20 colour raster graphics device was used to present graphical output interactively, employing a 3-d solids rendering package developed at the University of Victoria. Prolog was selected for a number of reasons, including: its suitability as a language for implementing expert systems [5], and its overt descriptive clarity.

3.1. The Need for an Expert Design System

Expert systems are powerful tools in knowledge-intensive fields of human expertise. Heuristics are often used to search problem spaces too large or heterogeneous for formal techniques.

As Alexander has observed [7], the number of potential interactions between rules in a generative system increases so explosively with the number of rules that a conventional exhaustive mathematical treatment of all points in the design space is impossible for non-trivial systems. He has also recommended [3,p.538] that designers should be free from preconceived notions, apply the pattern language objectively, and be egoless. Computer expert systems seem to meet these requirements.

3.2. The Knowledge-Base of Patterns

A subset of the patterns of Alexander's pattern language constitutes the expert knowledge of our system. The morphological content of patterns is represented in part by Prolog axioms, and can be manipulated (displayed, changed, removed, summarized) within the expert system. The solution component of each pattern is represented through the procedural attachment of appropriate Prolog routines to the pattern. Knowledge possessed by the system is thus both declarative and procedural in nature. The declarative knowledge indicates when and for what purpose the pattern is to be used. The procedural knowledge determines how the form being designed is affected when the pattern is applied.

Declarative information is represented for each pattern. The pattern number and name are present, to identify the pattern. Alexander's judgement of the universal archetypal validity of the pattern [2,p.xv] is represented by an integer from 0-2. The context of the pattern is encoded by associating, with each pattern, lists of the patterns immediately above and below it in the network of patterns. The group to which the pattern belongs is represented by an integer between 1 and 8. The problem solved by the pattern is given in text form. Finally, the solution component is given both in text form for explanatory purposes, and as (a set of) Prolog procedures.

The Prolog procedures for a pattern embody knowledge of the processing required to apply the design rule(s) captured by the pattern. Since Alexander intended his language to be used by people who possess spatial and perceptual tools (not easy to implement in machines), the amount of computation required to apply a even single pattern is often large. In our system, the complexity of individual rules (patterns) is much larger than that typical of production-based expert systems.

For example, consider Alexander's pattern number 106, termed 'positive outdoor space' [2]. This pattern solves the problem of unused spaces between buildings by making these spaces 'positive' in form, giving each some degree of enclosure. This intuitive notion corresponds closely to the mathematical idea of ensuring that the sum of areas enclosed by walls of buildings or segments which constitute the convex polygonal hull of the building's wings, weighted by the ratio of its enclosing perimeter (provided by the walls) to the entire perimeter of the hull, is sufficiently high.

Although the attachment of a plethora of procedures to a single pattern seems to violate the principle of modularization, it was necessary to create such complex rules in order to automate Alexander's system in its original form. The lower-level procedures typically employ geometric methods to solve technical problems which are simply taken for granted in human design.

3.3. Pattern Selection and Ordering

The first step in the design of a form is the selection of the patterns which are to be used in the design of the form. According to the instructions of Alexander [2,p.xxxviii], in order to design a form, one begins by selecting the single pattern which

'best describes the overall scope of the project [one] has in mind'.

The network of patterns is traversed forward from the position occupied by this initial pattern; all patterns below the initially-chosen pattern are presented to the user for selection or rejection. The user should select all patterns relevant to the form to be designed.

Alexander cautions the user against selecting irrelevant patterns:

'When in doubt about a pattern, don't include it. Your list can easily get too long; and it if it does, it will become confusing.' [2,p.xxxix]

One goal of the development of an expert system for design is to succeed in dealing with complexity without confusion. As the user traverses the network of pattern in the knowledge base, explanations can be given of the problems resolved by the patterns encountered, to help in deciding whether to include the pattern.

Following the selection of patterns, those chosen are sorted into the order in which they are to be applied, according to the heuristic rules of section 2.2.

3.4. Form Generation

At the beginning of the generation process, a number of global design parameters are requested of the user. For example, what is the square footage of the form to be designed? Although Alexander's patterns do not explicitly use such information, the creation of actual buildings requires it. Selected patterns are then used, one at a time, to differentiate the form.

The Prolog procedures attached to each of the selected patterns are then applied sequentially to differentiate the form. Prior to the application of each procedure, the user may chose to invoke one of several available options. If desired, an explanation of why the pattern is being applied is presented. At the end of the partial differentiation which results from application of a series of patterns from the same group, special post-processing procedures are invoked, to 'clean-up' the form which results from the processing.

A graphical display of the form as it currently exists at a given stage in the design procedure is updated as the form is differentiated by the application of patterns. As design proceeds, the user may request explanations of a design step or request the retention of the design for subsequent reproduction.

At the earlier stages of differentiation, the building is represented as a 2-dimensional line drawing in plan view. After all patterns from the first group have been applied, the form is given elevation, and is rendered thereafter by perspective projection of the 3-dimensional model. Through the scene description language employed by the rendering software, the user is free to select viewpoint, lighting sources, surface colours and reflectance properties for the building and its background both during and subsequent to the design.

3.5. An Example of Design

The following example of form generation is intended to convey the level of design expertise currently present in our system. Nine patterns [2] were selected for use in the design:

104) Site Repair
105) South-facing Outdoors
106) Positive Outdoor Space
107) Wings of Light
108) Connected Buildings
117) Sheltering Roof
128) Indoor Sunlight
159) Light on Two Sides of Every Room
239) Small Panes

Pattern 104 prohibits the form from being generated in certain regions of the site, as buildings should be constructed on those parts of the land which are in the worst condition. Pattern 105 establishes the principal orientation of the building. Graphical output commences for this example with the application of pattern 107, which causes the building to be generated in the form of long narrow wings (Figure 1).

Pattern 106, which occurs below 107 in the pattern hierarchy, is applied next, to ensure that sufficient space is enclosed by the proposed form (Figure 2). Pattern 108 ensures that the building is connected, and pushes wings towards their center of gravity if any were disconnected. As by chance all wings were connected, the application of this pattern causes no change in the example design.

As the next pattern to be applied, 117, is in a different group than is 108, end-of-group processing for group 1 causes extension of the building into the second and third dimensions (Figures 3,4). Pattern 117 causes the roof to be partially differentiated (Figure 5), and the remaining patterns create and differentiate windows (Figure 6). Figures 7 through 10 provide some views of the completed form.

4. Conclusions

Although our design system is still being developed, our results are promising, and convince us of the feasibility of a Prolog-based expert system which implements the pattern language of Christopher Alexander.

The usefulness of the system as a practical design tool is reduced by a number of factors. Alexander's design methodology, although formal and precise in many of its higher-level characteristics, requires the user to make many lower-level decisions based on intuition alone. That is, the language is morphologically complete in its specification of form only to a certain level of detail, beyond which it seems natural and simple for humans to continue on the basis of considerations which appeal to emotional feeling.

While the mechanisms for choosing patterns and applying the language are easily implementable, it is difficult to automate the entire process of form generation, because the intuitive feelings of human designers are difficult to characterize formally in expert systems. It is only due to the high degree of formal structure present in Alexander's system that the automation of design is conceivable at all.

We believe that introspection combined with experimental verification can in all cases reveal the formal substance of what is termed 'feeling' or 'intuition' about design. The production of a formal characterization of human feelings about design can in many cases be laborious; its difficulty was greatly underestimated at the commencement of our experiment.

Another problem encountered during the development of the system was the lack of a good programming environment for Prolog. In order to develop a convenient user interface for the system, a large amount of low-level Prolog code had to be written. Prolog routines for such tasks as list manipulation, I/O, screen management, menu presentation and raster graphics have all been created, and are now available for future projects and the continuing development of our system[3].

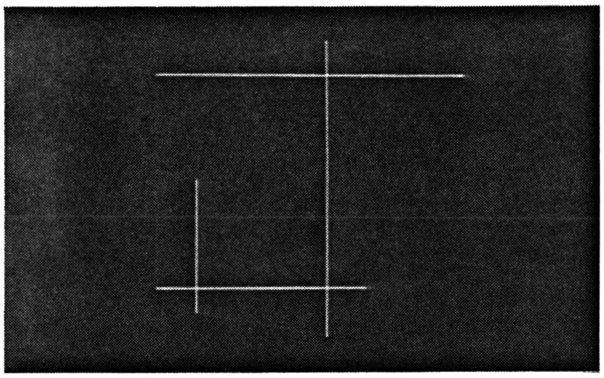
Our experiences with Alexander's design methods leave us optimistic about the future of complex knowledge-based expert systems in the field of architectural design.

[3]The Prolog source code for the system is available from the authors upon request.

References

1. C. Alexander, "The Bead Game Conjecture," *Lotus, an International Review of Contemporary Architecture* **5** pp. 151-154 Fantoni Artegrafica, (1968).
2. C. Alexander, *A Pattern Language,* Oxford University Press, New York (1977).
3. C. Alexander, *The Timeless Way of Building,* Oxford University Press, New York (1979).
4. N. Chomsky, *Aspects of the Theory of Syntax*1956.
5. K.L. Clark and F.G. McCabe, "PROLOG: A Language for Implementing Expert Systems," pp. 455-470 in *Machine Intelligence 10,* ed. D. Michie,John Wiley & Sons, New York (1982).
6. C.S. Clocksin and W.F. Mellish, *Programming in PROLOG,* Springer-Verlag, New York (1982).
7. S. Grabow, "The Science of Design: Christopher Alexander's Search for a Generative Structure," *ReVisions,* (2) pp. 36-45 (Fall, 1983).
8. R.A. Kowalski, "Predicate Logic as a Programming Language," *IFIP 74,* pp. 569-574 North-Holland, (1974).

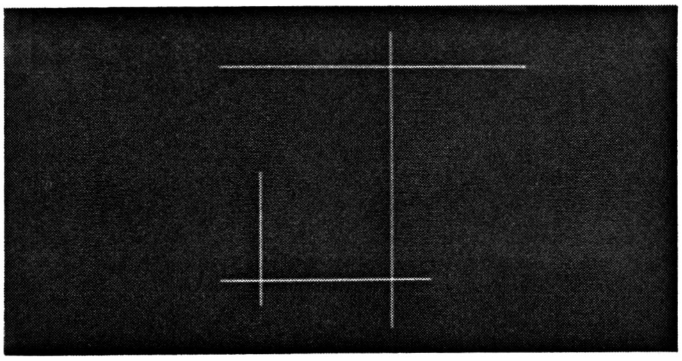
PLAN VIEW.SCALE 1:500



Pattern 106:Positive Outdoor Space

Figure 2

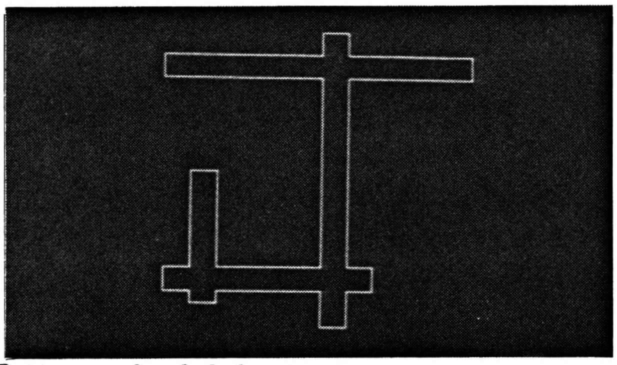
PLAN VIEW.SCALE 1:500



Pattern 107:Wings of Light

Figure 1

PLAN VIEW.SCALE 1:500



Pattern endgrpl:2-d extension

Figure 3

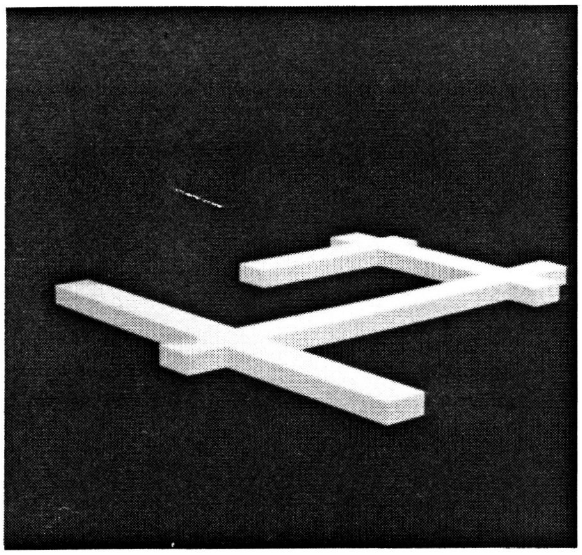


Figure 4

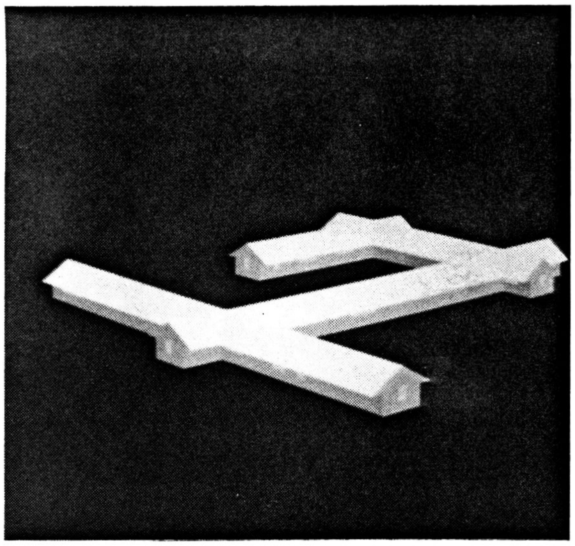


Figure 6

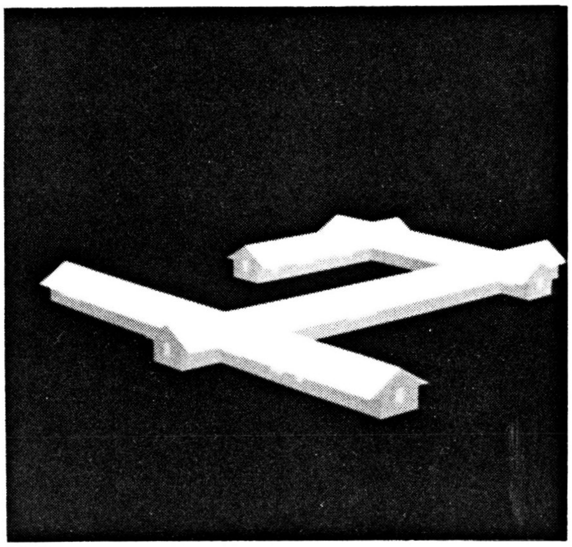


Figure 5

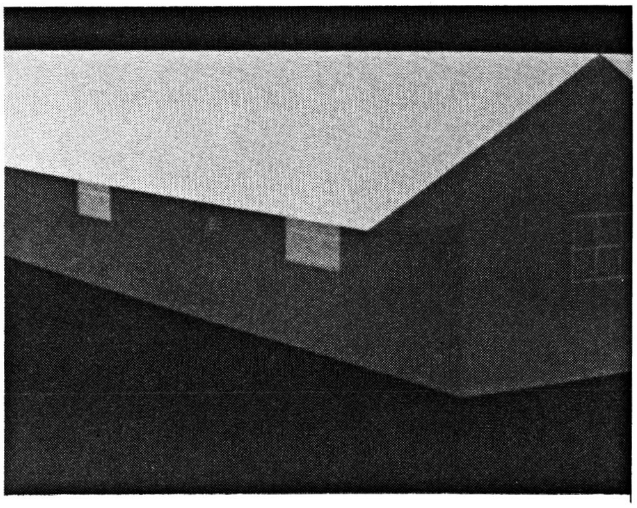


Figure 7

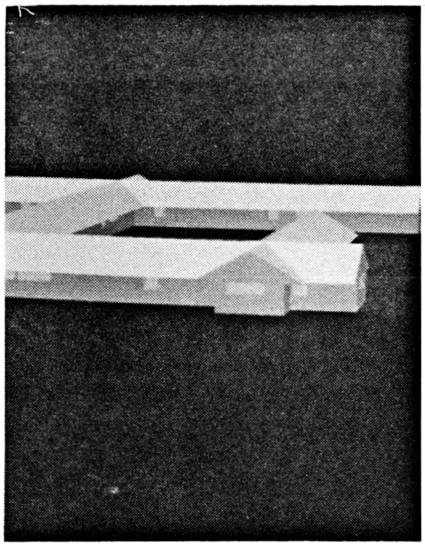


Figure 8

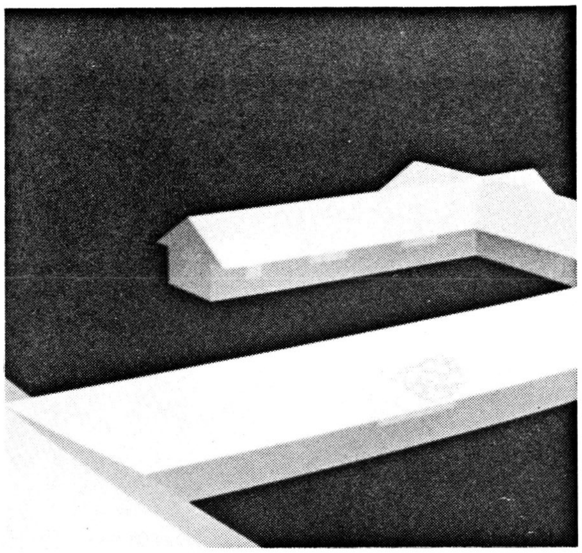


Figure 10

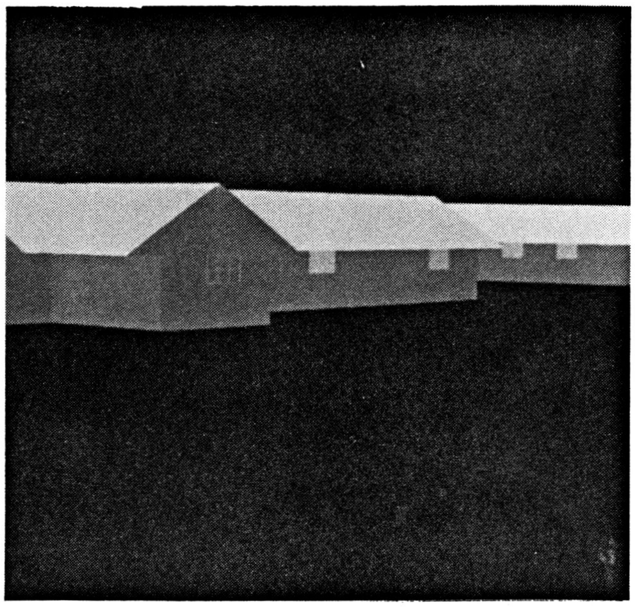


Figure 9