4
Computer Modeling:
A First Course in Design Computing

Bharat Dave
Robert Woodbury

Department of Architecture
Carnegie-Mellon University

Introduction

Computation in design has long been a focus in our department. In recent years our faculty has paid particular attention to the use of computation in professional architectural education. The result is a shared vision of computers in the curriculum Woodbury 19851 and a set of courses, some with considerable history[1] and others just now being initiated[2]. We (Dave and Woodbury)[3] have jointly developed and at various times over the last seven years have taught Computer Modeling, the most introductory of these courses. This is a required course for all the incoming freshmen students in the department. In this paper we describe Computer Modeling: its context, the issues and topics it addresses, the tasks it requires of students, and the questions and opportunities that it raises.

Computer Modeling is a course about concepts, about ways of explicitly understanding design and its relation to computation. Procedural skills and algorithmic problem solving techniques are given only secondary emphasis. In essential terms, the course is about models, of design processes, of designed objects, of computation and of computational design. Its lessons are intended to communicate a structure of such models to students and through this structure to demonstrate a relationship between computation and design. It is hoped that this structure can be used as a framework, around which students can continue to develop an understanding of computers in design.

Intellectual Underpinnings

Underlying the course is the notion of a model, which Echenique [Echenique 1972, p.1641 aptly characterizes:

"...a representation of a reality, in which the representation is made by the expression of *certain relevant characteristics of the observed reality and where reality consists of objects or systems that exist, have existed, or may exist.'*

We present the entire course as a conscious exercise in model building (modeling) and explanation. We feel that it is important to present a single coherent view, and we thus have to embrace a single overall model. We find that the notion of rational decision making (RDM) has much to commend itself. According to this notion, design is an iterative, or cyclical, activity with distinct types of actions occurring in a fixed order in each cycle. These actions are: the generation of alternatives, performance prediction, the simultaneous evaluation of predictions, and the selection of alternatives. Further the process is purposeful, it has goals, and these are captured in the evaluation and selection steps. Stepping outside of this local view of an individual decision cycle, the basic model can be seen to accommodate a wide variety of global processes, from strictly linear ones in which only one alternative is generated at each step and no changes are ever made to decisions already made, to very complex processes, with multiple agents, with compound branching and significant retraction of decisions.

All of the activities involved in rational decision making must he carried out by some agent and further structure is introduced when that agent is human. To describe this structure we introduce as the second of our models an abstract model of human cognition [Newell 19721. According to this model, the structure of the human mind itself necessitates a certain way of conducting design. Humans design by searching in a state space; they move from one representation (or model) of a designed object to another, at each step applying the last three actions of the RDM cycle to some degree. Further, humans have a very limited capacity for immediate storage and access of information, and this implies that they will virtually always use some form of externally stored information to assist them in designing. This information may represent the object being designed, the process of design, the goals of design or general knowledge about objects or designing.

Humans code, and understand, all of this information as symbol structures, that is, sets of discrete objects and relations between them. This way of representing objects works well in many contexts but encounters difficulties in architectural design. Space, a continuous phenomenon, is at the center of everything in architecture and its discrete representation has always been

problematic. Over the centuries a number of ways of representing things spatial, none entirely satisfactory, have evolved in architecture, and the current intellectual struggles in building computer based spatial representations continue this pattern. We explain the relation between representation (in the form of symbol structures) and space as the third model of the course. To demonstrate this idea, and to introduce them in their own right, we use examples of spatial representations as the basis for sections of the course.

The three models above, rational decision making, human cognition, and spatial representation, describe design as done by humans. Our course is about computation in design and two additional models must be brought to the table. The first is a model of computation, that describes single processor serial machines, operating on dual memory, The second is an augmentation *of* the basic model *of* human cognition to include these computational agents. This last model is the only normative and speculative model in the course. In its presentation we take the position that the goal of using computation in design is to be supportive of humans in areas of strength and to provide new capabilities in areas of human weakness.

Computer Modeling thus consists of explaining five models. Taken together we believe that these constitute a framework on which a broad understanding of design computation can be built. Through explanation, demonstration, and a set of structured exercises we hope to help students learn and apply these models. Since our goal is communication of the ideas, we stress a conceptual approach in the course- Students are given the concepts in lectures and a minimal amount of procedural instruction in labs. The assignments are designed to stress the concepts and to exercise the students use of the models. By and large, the procedural parts of the assignments are left in the students' hands.

Course Content

We structure the course around examples of spatial representations (or models); introducing and explaining the other four models in the context of specific representations. Each of these-graphic pictures, graphic *2-dimensional, graphic 3-dimensional, databases, and* parametric models-represents and supports manipulation of design information in a different fashion, thus providing ample grist for our pedagogical mill. For each spatial type, we provide a set of lectures and computer based exercises using existing commercial software.

Graphic Pictures

Graphic pictures (our name in this course for bitmaps) are collections of point marks; these are perceived by humans to possess some properties like pattern, orientation, and so on. Our tendency to see some organizing pattern (or lack

thereof) lets us associate representations of points with larger conceptual elements like lines, surfaces, texture, and others. Thus graphic pictures as models make use of points as basic building blocks, and patterns of points to represent semantic concepts. A set of fundamental and most often-used operations on picture-based models are the similarity transformations: translation, rotation, scaling, and reflection. An exercise to introduce these notions is to alter (by subdivision, line editing, introduction of elements, etc.) a polygon in such a way that when many such polygons are packed together using the similarity transformations, an aesthetically pleasing and complex composition is obtained (figure 1).

Within this exercise, some of the other course models are easy to introduce. The exercise is stated in terms of an initial state (the polygon), and a suggested set of operators upon that state (subdivision, line editing, introduction of new elements, etc.). We have observed that successful students naturally use a state space approach in this context. We elaborate on state space search here to introduce the notion of a rule, a semantically meaningful operation, and show that rule discovery and search are two inseparable aspects of state space search. Further, each alteration to a polygon has often unexpected effects on a composition of many polygons; at the end of a set of operator applications students usually find it necessary to stop and compose a few polygons together to

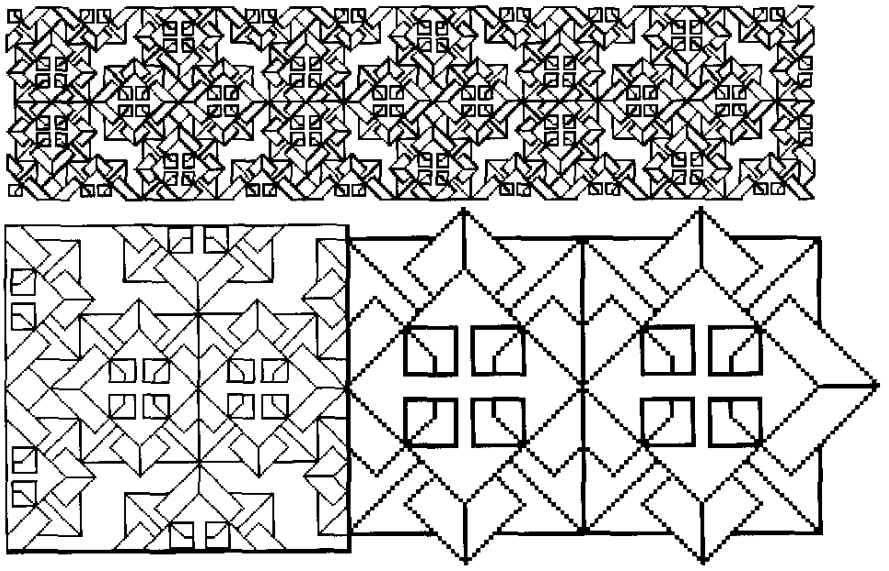

Figure 1 Paint system exercise (Michael Choung)

evaluate the overall effect. This provides a natural opportunity to discuss the RDM model.

We use paint programs as computational tools to develop graphic pictures. The painting paradigm operates directly on a bitmap, and input from the bitmap to any operators must be stated in terms of pixels and (possibly ordered) sets of pixels only. The operations may appear to output higher level entities such as lines and polygons, but these are laid into the bitmap representations, and their meaning is immediately lost in the representation, though often not in perception of that representation by humans. We demonstrate to students that painting thus supports only a small part of the operations required for the task, and that with painting, rule creation and application must reside entirely on their side of human computer interaction. At this point, we also introduce the notion of a procedure, or an ordered series of operations, by showing that many of the operations required in the exercise occur in repetitive groups. As manually executing these repetitive groups is the most oppressive part of the task, a strong motivation for understanding and using procedures is created.

Graphic 2D Models

If we model and operate upon representations of objects as collections of points only, the amount of information to be represented and processed becomes unwieldy. By adding certain structure to graphic pictures, we obtain graphic 2D models. A set of graphic 20 elements like lines, curves, etc. and a set of operations upon them are defined in graphic 20 models. A point represents a location in space and a certain order of distribution of points is used as a basis for construction of graphic 2D elements. For example, given a center and a radius, a circle represents locations of all points such that all points are equidistant from the center by an amount equal to the radius. A set of points are coalesced into a small number of parameters to completely define all points in that set. Graphic 20 models thus make use of geometric concepts and when applied in a domain like architectural design gain further semantic structure by concepts and conventions of that domain.

These notions are introduced in an exercise in which students design an interior layout for a given site and develop a set of scaled drawings. This exercise is an elaboration of some ideas presented in the earlier section. The goal is to develop a spatial composition using a set of familiar objects and their relationships. Unlike developing a graphic composition as an end in itself as in the earlier exercise, students have to take into account architectural requirements of the spatial composition. The iterative process of design development as RDM and state space search can be made more explicit by requiring students to record intermediate design layouts and sketches.

In this exercise, scaled drawings are used as the means to represent and conduct design inquiry. In particular, the role of representations and inferences they support are emphasized. For example, a square may be represented once at least the length of its side is known. Even if we do not know where it is located, we can infer its area since it is defined in terms of parameters we already know. On the other hand, if we require a square of certain area, we can infer the length of its sides. This process of translating known information into other desired forms enables us to represent and communicate our ideas, and this very process also allows us to translate design purposes into realizable products.

By using 20 drawing programs in this context, we introduce additional computational notions. While working on a traditional medium like paper, articulation of design intentions and their realization as a design solution are the primary responsibility of human agents. In contrast, while working on computers, students have to not only articulate their intentions but also explicitly select and communicate such operations to a drawing program. This observation leads to further discussion of structured data representation, i.e. from pixels to graphic 2D elements and attributes, and representation of actions as procedures or parametric operations, e.g. creation of a square by specification of two opposite vertices. By seeing and realizing that interpretation and execution of purposeful actions as procedures can be encoded in computer programs, students are exposed to further possibilities of computers augmenting the design process.

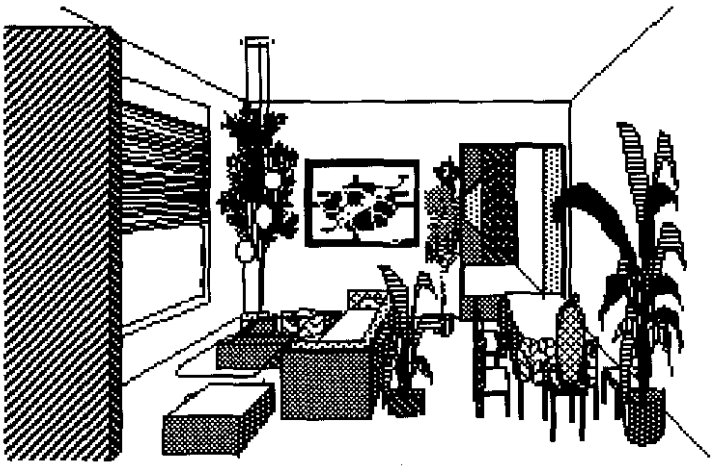

Figure 2 2D Drawing Systems (Yi-Ren Wang)

Graphic 3D Models

Two dimensional graphical *representations like* plans, sections, etc., are but partial views of spatial environments, To make *these views* as informationally consistent and *complete with* real spatial environments as possible, we introduce graphic 3D models. Like the way in which point-based graphic pictures *were extended* to define higher level graphic 2D models, we extend graphic 2D *elements into* the third dimension by introducing the notion of volumes that can he solids or voids. Based on certain *parameters, volumes* can be *categorized into classes, e.g.* cubes, cylinders, spheres, and others. By using *these parameters, a* small number of parametric definitions can be used to *generate and* represent a large number of instances of volumes.

Application of graphic 3D models in a domain like architectural design *requires introduction* of additional parameters-domain specific attributes and relations-to *these models*. For example, an instance of a cube may represent a wall made out of *concrete blocks* of certain strength and color. Another instance of a cube may *represent void* space with qualitative attributes like acoustic and lighting *properties*.

We introduce *these ideas* using an exercise in which students develop a graphic 3D model (at a certain *level of* detail) of an existing design project. In this *exercise, the* emphasis is not on the generation of a new design but rather on its *representation in* terms of design elements. By taking apart and modeling a design project, students are expected to understand design projects as *assemblies of physical and qualitative elements brought* together in order to satisfy some design context. The underlying model of design as a set of purposes and their realization in the form of volumetric compositions *becomes more* articulated in this *exercise. Further,* parametric relations in graphic 3D models can be used to infer new information-both geometric and architectural. Thus, given a cube with its length, width and height, we can infer its volume; and conversely, given a volume and some constraints on any two other *parameters, the* third can be inferred. Similarly, architectural concepts like adjacency, spatial continuity, rhythm, etc. are *realized as* well as evaluated using volumetric *parameters and* relations between them.

The role of 3D drawing programs in this *exercise is* to make obvious the issues discussed above. Further, students also *realize that* the underlying volumetric model in a modeling program affects the *level of* detail and consistency of information that can be *represented. Thus,* a 3D drawing program based on constructive solid *geometry exhibits* different capabilities than the one based on *sweep representation or* octrees. Additionally, data representation

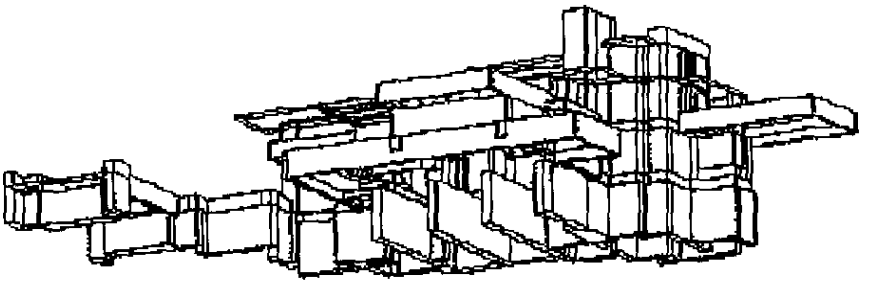

Figure 3 3D Drawing system exercise (Glenn Cottrell and Matt Wittmer)

and operations supported in such programs are organized as one large set of structured information-a database. Various views like plans, elevations, perspectives, are results of operations that retrieve and transform selected information from that database. What we understand and represent about a phenomenon is affected by the model we use and the vantage point we choose. On a secondary level, computer programs that support 3D representation, hidden-line removal, and specification of attributes like color, texture, light sources, etc., also enable accurate visualization of three dimensional spaces. This has a subtle impact on students in that they become more attuned to thinking in three dimensions since such programs also offer fast generation of realistic 3D views.

Databases

The graphic models introduced earlier are sufficient to model a number of phases in architectural design. These models are made more complete by introducing the notion of databases that integrate graphic and nongraphic representations. Architectural design projects start with a statement of needs or a design brief. Based on a programmatic description, a design is developed using various kinds of graphic and nongraphic representations, e.g. architectural and engineering drawings, specifications, bills of quantities, and others. Once a project is constructed, another set of descriptions may he developed to record the design project as built. Taken together, all such project descriptions constitute a project database in which each description is derived based on another description and augments or selectively transforms information.

An exercise introduces these notions. The task is to develop a database of project information for interior layout for a given site. Some of the notions about the model of design as the RDM and state space search,

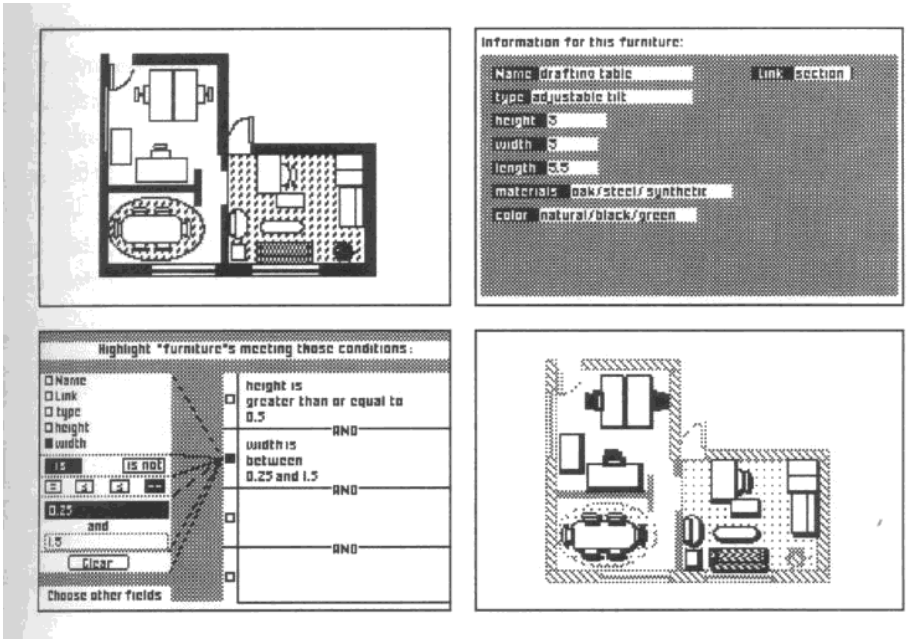

Figure 4 Database exercise (Jeff Goldstone)

introduced earlier remain the same in this exercise. Additionally, students learn to select a complete set of parameters-both the graphic and nongraphic, that are necessary to develop a model of spatial entities.

Design databases offer two significant advantages: generation of new descriptions as needed and reduction of effort in representing those aspects of design that are repetitive within a project and across a number of projects. In the former case, that is generation of a new description, we take a given description, define a format for new description and extract information from what is available. For example, given a plan, a schedule of doors and windows for each opening represented in drawings can be generated. This is an example of using information that is implicit in a graphic representation to derive a nongraphic representation based on some selection criteria. In addition, the newly derived description may be used with other graphic or nongraphic information to generate yet another description. To illustrate, a drawing may explicitly describe only a tiling pattern and material specifications for some floor area. Based on this, a total required number of tiles may be derived. Using this quantitative description and construction costs, a new description, e.g. cost estimate, can be derived. In other words, a design database consists of a number of graphic and nongraphic descriptions that are related, and using some operations, a given description can be transformed into another description.

The second advantage, that is reduction in effort, derives from the fact that certain design information may be used quite often in a design project, (e.g. window details), or a number of projects may use similar details, (e.g. elevator cores). Such information that does not change and is used repetitively can be stored in a standard details database, and can be referenced from that database when needed.

These notions are made more explicit by using a computer-based database program. In order to develop a project database, students have to define, represent and associate a sufficient number of parameters in both graphic and nongraphic descriptions. *Further, operations* of sorting and searching on the information represented give students an opportunity to selectively generate new descriptions. This also leads to the discussion of some parallels between database concepts and design experience. A database is a collection of structured information, and such information can be collated in various ways to generate new information relative to design goals. Similarly, design expertise can be viewed as a large collection of design rules that are applied in varying contexts, and their combined processing brings forth new design descriptions.

Parametric Variations

According to the model of design as RDM and state space search, design development goes through a number of stages; at each stage, some information is *asserted in* keeping with given design purposes. Each stage in the design process supplements and transforms information from the preceding stage. Throughout this process, design *representations are* externalized in the form of drawings, sketches, and others. Externalization of representations consists of asserting relevant information about design elements and some techniques or procedures to manipulate and transform such data.

In the process of generation and searching for appropriate representations, the critical issues are selection of operators and statement of a final goal that controls and directs the sequence of design information processing. But in most design tasks, information states and operators are not so clear cut. Information available at the start of design process may be only loosely or partially defined. At any given stage in the design process, more than one operator may be available to manipulate and transform a current design state. Out of a huge amount of information, a designer has to select appropriate transformations and an order in which they are to be carried out. This situation in design leads to a large number of possible alternatives which are but variations derived by changing some design parameters. This process of design exploration can be viewed as generating various design scenarios.

These concepts are introduced using an exercise in which students build a parametric model for calculating illumination levels in a space given some

artificial light source. Once a parametric model is defined, students generate various design alternatives by either changing the location of light source, its intensity, or both. The results are graphically displayed using graphs and contour charts.

This exercise, in a limited fashion, brings together a number of issues as they pertain to modeling in architecture and computation. By explicitly developing a parametric model of a design task and generating a number of alternatives, students learn to articulate dependencies among design parameters. Design scenarios may involve parameters that are independent of each other, e.g. placement of a stairwell and floor finishes are relatively independent of each other. On the other hand, both the number and size of steps are dependent on floor height and linear distance available in a given design. The latter kinds of decisions involve relational information involving dependent parameters. If the significant parameter is tweaked, its effect will propagate to parameters that are dependent on it. Such relational information can be modeled as algebraic equations, e.g. computing number and size of steps given other relevant information.

Secondly, this exercise also gives students a chance to explicitly define a state space, and search for feasible states for a given design task. By using spreadsheet programs, the ideas of computation as representation of data, relations between them, and procedures by which possibilities inherent in a model can be explored, are made obvious. This, in turn, leads to the discussion of parallels between architectural design process and computation, and how the experience accumulated by designers over a number of years might be represented and used in a computer. The most significant impact of experience manifests itself in the selectivity exercised in design development. The kinds and ways in which experienced designers generate and manipulate design information evolves into a consistent pattern. This consistency of design development or information processing in design is what gives rise to identifiable design styles. These ideas become evident once we explain a paradigm like shape grammars and their application to the works of some noted architects.

Course Summary

The ideas introduced in the course exhibit a tight hierarchy in that each exercise is an enhanced and more structured version of the one preceding it. Once students have completed these exercises, we present a taxonomy of computerassisted design paradigms (Galle 1981) and representative examples for each of the following categories:

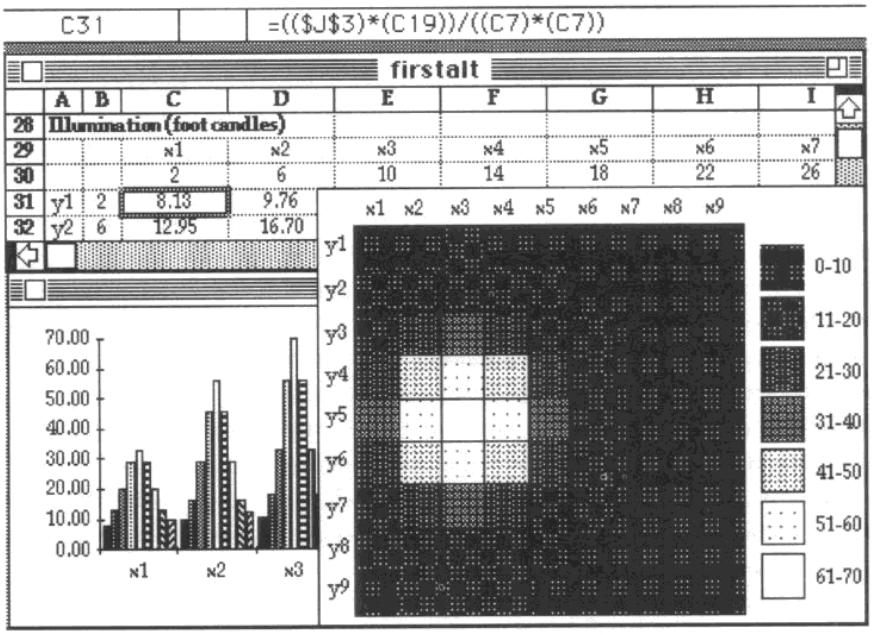

Figure 5 Spreadsheet exercise (Christine Kochinski)

1. Automated evaluation of designs generated by traditional means
2. Stepwise interactive generation
3. Non-exhaustive generation of feasible solutions
4. Exhaustive generation of feasible solutions
5. Generation of (sub)optimal designs

By taking a structured approach to the introduction of these ideas, we hope that students acquire a conceptual understanding of the role of models and their representations. By emphasizing a conceptual over a procedural approach, we also hope that students articulate the knowledge they gain in this course in a fashion that is independent of particular tasks, domains, and software or hardware considerations.

Course Materials and Facilities

The course is organized as a combination of lecture sessions, alternating with hands-on demonstration in the use of computers and various applications. Lecture sessions are scheduled for about 50 minutes, once every week. These sessions are used to introduce concepts, and are supplemented with a set of recommended readings. Two texts that we have found useful in this regard are Mitchell 19771 and (Radford 1987). We have also developed an extensive set of

lecture notes that closely follows the course structure, and that is made available to students.

Lab sessions meet for about 80 minutes, once every week. Currently, we use MacPaint[4], MacDraw[5], FileVision[6], and Excel[7] running on the Apple Macintosh[8] machines to introduce paint systems, two-dimensional drawing systems, databases, and spreadsheets. For three-dimensional drawing systems, we use AutoCad[9] running on the IBM AT[10] machines. All these machines, software applications, and printing facilities are maintained in public clusters on campus and are accessible to students on an extended basis including weekends.

Reflections

An informal evaluation of the course through student feedback, both written and oral, and course assignments, gives us a general feeling of satisfaction with both format and content. We are particularly pleased with our choice to use representation as a common thread throughout the course and to explain the other four models within this context. When we designed the course, we saw this format as the one that provided the most concrete basis from which to build-a view we still hold today. We have found that students are increasingly interested in learning about computers and their architectural applications. The course seems to provide a good beginning for these interests; indeed, some students, after taking the course, have spontaneously used their experience in other courses.

These preenings aside, we see several areas that can be improved. Each of our course components could build more structured links to the five basic models we communicate, and to other courses in the departmental sequence The graphic picture (bitmap) component could more formally introduce geometric operations and could be more explicit about state space search. In the 2D model component, we get relatively small intellectual content from a large amount of student drawing effort. Neither the particular topic (interior layout) nor the drawings required (plan, section, and perspective) provide any especially stimulating content to the exercise. In the 3D model component, we are somewhat frustrated by the lack of appropriate solid modeling systems on inexpensive hardware. The content of the database component has improved recently, but it remains at too trivial a level to really motivate student interest. Finally, the spreadsheet module is very interesting, but we have just begun to explore the possibilities for architecturally relevant exercises.

On a larger scale, we feel a need to more tightly integrate Computer Modeling with the rest of our course sequence and with the departmental curriculum as a whole, while maintaining its emphasis on concepts independent of particular applications and tasks. A stronger treatment of procedural ideas

would assist our introductory and advanced programming courses. More rigorous consideration of geometry could motivate our course on geometry and computation. If search were treated more seriously, students could be better prepared for the material in our advanced course on rules and representations.

References

[Echenique 1972] M. Echenique. 'Models: A discussion." Chapter 7, pages 164-174. In L. Martin and L. March (eds.), *Urban Space and Structures,* Cambridge: Cambridge University Press.

[Galle 1981] P. Galle. 'An algorithm for exhaustive generation of building floor plans." Pages 813-825. Communications of *the A*CM, 24, 1981.

[Mitchell 1977] W. Mitchell. *Computer-Aided Architectural Design*. New York: Van Nostrand Reinhold.

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

[Radford 1987] A. Radford and C. Stevens. CAAD *Made Easy*. New York: McGraw Hill.

[Woodbury 1985] U. Flemming, 0. Akin, It Woodbury. "A Comprehensive Computing System for Architectural Education". pp 637-642. In K. Duncan and D. Harris (eds.), Computers in *Education, First* World Conference on Computers in *Education. Amsterdam:* Elsevier Science Publishers By. (North-Holland).

Notes

1 Introduction to Programming, Introduction to CAD, Rules and Representations, CAD Laboratory

2 Geometry and Computation, Rational Decision Making

3 Roy McKelvey, faculty in the Department of Design, has also played a significant role in teaching Computer Modeling.

4 MacPaint is a product of Claris Software.

5 MacDraw is a product of Claris Software.

6 FileVision is a product of Telos Software.

7 Excel is a product of Microsoft Coropration.

8 Macintosh is a product of Apple Computers.

9 AutoCad is a product of Autodesk.

10 Personal computer AT is a product of IBM Corporation.