Modeling Architectural Design Objectives in Physically Based Space Planning

Scott A. Arvin and **Donald H. House**
Texas A&M University, USA

Physically based space planning is a means for automating the conceptual design process by applying the physics of motion to space plan elements. This methodology provides for a responsive design process, which allows a designer to easily make decisions whose consequences immediately propagate throughout the design. It combines the speed of automated design methods with the flexibility of manual design methods, while adding a highly interactive quality and a sense of collaboration with the design itself.

In our approach, the designer creates a space plan by specifying and modifying graphic design objectives rather than by directly manipulating primitive geometry. The plan adapts to the changing state of objectives by applying the physics of motion to its elements.

For design objectives to have an effect on a physically based space plan, they need to be able to apply appropriate forces to space plan elements. Space planning can be separated into two problems, determining topological properties and determining geometric properties. Design objectives can then be categorized as topological or geometric objectives. Topological objectives influence the location of individual spaces, affecting how one space relates to another. Geometric objectives influence the size and shape of space boundaries, affecting the dimensions of individual walls.

This paper focuses on how to model a variety of design objectives for use in a physically based space planning system. We describe how topological objectives, such as adjacency and orientation, can be modeled to apply forces to space locations, and how geometric objectives, such as area, proportion, and alignment, can be modeled to apply forces to boundary edges.

Introduction

As we envision it, *responsive design* is a process that allows a designer to easily make decisions whose consequences immediately propagate throughout the design. Such a responsive design process would be automated, natural, intuitive, flexible, and interactive. It would be automated in that given a set of design intentions, a design solution can be produced. It would be natural in that design elements and intentions are specified in the working language of the designer rather than a low-level representation. It would be intuitive in that the individual elements and the overall design act in expected ways. It would be flexible in that designs are easily modifiable and the design process does not specify the "best" design, but enables designs to emerge. Finally, it would be interactive in that responses to user inputs happen in real time. The key to a responsive design system is to define a powerful set of design objectives that enable the designer to think in higher-level design terms rather than lower level geometric terms.

We are attempting to create a methodology for a responsive design process by applying physically based techniques to architectural space planning. Although we believe that responsive design can be applied to many graphic design domains, space planning is a relatively simple and well-studied area in which to begin to understand and apply the complexities of this methodology.

This paper focuses on modeling design objectives as force objects used in a physically based space planning system. We will review the overall structure and implementation of our approach described in Arvin and House (1999). There we originally defined the term *design* objective, but described the implementation of only two basic design objectives. Here we present implementations for modeling a wider variety of design objectives, and demonstrate how they can be applied to the space planning design process.

In our approach to space planning the architect defines programmatic objectives in the usual manner, and these objectives are then modeled as physical objects and forces used in a dynamic physical simulation. We model spaces and walls as point masses, with adjacencies between spaces modeled as springs connecting the masses. Objectives specified in the architectural program are translated into forces applied to the masses. A dynamic simulation proceeds allowing the mass-spring system to quickly reach equilibrium. The designer then modifies and adds objectives by directly manipulating the graphic model rather than by respecifying design objectives in the language of the underlying system. The mass-spring representation allows the graphic model to adapt to those changes immediately. We do not intend to simulate the actual behavior of building elements, but to simulate the way architects may view and interact with design elements during their conception.

Preliminary results of using physically based design objectives on simple floor plans indicate that this responsive approach is extremely promising. We have to date been focusing our efforts on a pilot study, developing this approach for single story, rectangular spaces. But even on simple models it feels natural and fun to add and modify physically based design objectives to try out many designs. The ease and speed of generating and manipulating designs allow new and more interesting ideas to emerge that would be more difficult to discover with non-responsive approaches.

Background

There exists a wide body of research into automated space planning methods. We present here just a few of those methods that contrast with ours. We then describe the concept of physically based modeling and review some of basic components used in physically based simulations.

Automated space planning

Architectural space layout problems tend to be ill-defined (Yoon, 1992, p. 8) and over-constrained. Problems that are not well defined are ill-defined (Simon, 1973), in that the initial constraints on the problem are not fully formulated. Resolving ill-defined problems is a process of

searching for and refining a set of design constraints. Problems that are over-constrained have no single best solution and thus have too many possible solutions (Balachandran and Gero, 1987). Automated space planning systems need a method of providing a good solution from a large set of possible solutions, and a method of allowing the designer to modify the set of design constraints to continually refine the problem definition.

Some approaches to automated space layout planning use an iterative process of constructive initial placement, in which spaces are positioned one at a time (Liggett and Mitchell, 1981; Flemming and Chien, 1995). In these approaches, an ordering function is needed to determine which space to position first.

Some approaches are generative in nature, in that they seek to produce all or a large number of the possible designs within a design space. Some examples are evolutionary design techniques (Jo and Gero, 1998; Gero and Kazakov, 1998), and shape grammars (Flemming, 1987).

Methods of producing optimal space plans have been the focus of many approaches such as Liggett and Mitchell (1981).

Weinzapfel and Handel (1975) describe an approach to automated space planning in which a design problem consists of a set of spaces and a set of relationships, similar to our design objectives, describing constraints on the spaces. Their solution routine repeatedly iterates through each space, testing all relationships that affect a space to determine the next location that best meets those relationships. Conflicting relationships are solved using an optimization technique called Least Mean Squares Fit. A drawback of their approach is that the order in which the spaces are evaluated affects the final solution.

Constraints have been used in architectural design (Gross et. al., 1986) in, for example, three-dimensional solid modeling (Tobin, 1991; Martini, 1995), and space layout planning (Yoon, 1992). Others have described a variety of design constraint types such as dimensional, ratio, adjacency, orientation, and shape constraints (Mitchell 1977, p. 430; Pfefferkorn, 1975).

One of the most notable and extensive research projects in recent years is the "Software Environment to support the Early phases in building Design," or SEED (Flemming and Woodbury, 1995). SEED partitions the schematic design problem into a variety of modules, one of which is SEED-Layout (Flemming and Chien, 1995). SEED-Layout supports design space exploration through an iterative, constraint based approach that can be either manual or automated. It also supports a case-based approach where previous designs can be used to produce new designs.

Physically based modeling

Physically based modeling is a subfield of computer graphics and visualization. It attempts to represent dynamic motion and changes in geometry by modeling objects as mechanical elements that behave according to the laws of physics. Dynamics are most often derived by the use of forward numerical simulation over discrete time intervals. In a forward simulation, the system is moved from its state at the current time to its state at the next discrete time step, using forces to determine accelerations, and thus changes in velocity during the time step, and velocities to determine transla-tions. The process of making this forward extrapolation is called numerical integration. An excel-lent introduction to the concepts of physically based modeling and a practical guide to the implementation of these concepts in the computer is given in Witkin and Baraff (1997).

Physically based modeling has been used to model the realistic behavior of rigid bodies (Barzel and Barr, 1988; Baraff, 1989) as well as deformable models (Terzopoulos et. al., 1987). Others have recently begun to use dynamics in geometric design. Qin and Vemuri (1998) and Mandal, et. al. (1997) use physically based techniques to manipulate smooth surfaces of arbitrary topology interactively. In their approach, a user defines the points of an initial control mesh, which are manipulated by applying synthesized

forces until the desired shape is achieved.

Harada (1997) and Harada et. al. (1995) use a physically based approach to allow for discrete and continuous manipulation of generalized floor planning problems. The approach is demonstrated using architectural floor plans represented as rectangular dissections, as well as circuit board layout and page layout.

A key concept in physically based modeling needed later in the paper is the mass-spring-damper modeling element. Figure 1 shows a simple mass-spring-damper system consisting of two points with mass m_0 and m_1, connected with a spring with spring constant k_{01} and a dashpot with damping constant d_{01}, and predefined with a desired rest length r_{01}. The current length l_{01} of the spring is the magnitude of the vector between the positions x_0 and x_1 at the current time. The spring exerts forces f_0 and f_1 with magnitude proportional (with proportionality constant k_{01}) to l_{01} - r_{01}. The direction of the force will be along the line connecting the point masses. As the masses move farther from each other, the spring applies a force to try to move them closer, and as they move closer the spring tries to separate them. The dashpot is attached in parallel with the spring, and acts like the hydrau-lic piston on a screen door closer. It damps the motion of the masses by producing forces propor-tional (with proportionality constant d_{01}) to their relative velocity towards or away from each other, thus reducing the kinetic energy introduced by the spring forces.

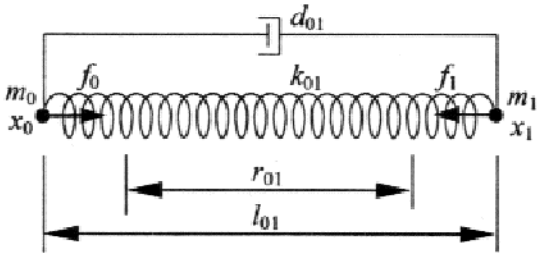

Figure 1. A mass-spring-damper system.

Problem

Given a set of elements that represent spaces and shapes in a physically based space planning system, how can we model design objectives to apply appropriate forces to these elements such that design goals can be achieved? We define a *design objective* as something that a designer wants to happen, specifically, any intention that influences geometry. Given a set of defined spaces, each with or without a defined boundary shape, a set of design objectives determines how a space planning solution will be resolved.

The space planning problem itself can be broken down into two sub-problems: determining topological or qualitative properties, and determining geometric or quantitative properties (Jo and Gero, 1998; Flemming, 1989). The topological problem is one of determining the relationship between individual spaces, without regard to the dimensions of any building elements. The geometric problem is one of determining the physical dimensions of the building elements. Both of these problems need to be solved in any approach to space planning.

In physically based space planning, design objectives need to be translated into forces that act on nodes. We separate them into two categories, topological objectives and geometric objectives, in keeping with the sub-problems noted above. Topological objectives apply forces to the center of a space, and geometric objectives apply forces to the polygonal edges of space boundaries. It is our working hypothesis that any design objective that affects the position of any element in an architectural plan can be translated into a force object.

We use the term *objective* instead of *constraint* to avoid confusion with the term used in physically based constrained dynamics. In constrained dynamics, a constraint is a condition that, once met, continues to be met throughout a dynamic simulation (Barzel and Barr, 1988; Witkin and Kass, 1989). One example is an area constraint, where any applied forces that may act to change the area of a polygon will be counteracted to ensure that the area remains

the same. Constrained dynamics is not a process of *constraint satisfaction*, but one of *constraint maintenance*. If an area is not as it should be, constrained dynamics will not act to change it to the correct area.

Implementation

In this section we review our fundamental physically based space planning elements, and then describe how we have implemented a variety of topological and geometric design objectives using these elements. Figure 2 summarizes the elements that will be described. A single space with center point node c is bounded by polygon *p*. The dots on *p*'s edges are line nodes used to define the edges. Arrows labeled with *a*'s represent forces applied by topological objectives to the space location, which may change the way this space relates to another. Arrows labeled with *b*'s represent forces applied by geometric objectives to the polygonal edges of the space boundary, which may change the geometric position of the edges.

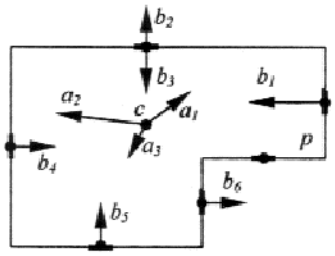

Figure 2. Simple space with forces acting on its elements.

Physically based space planning elements

The basic elements of a physically base space planning system are nodes, which define the position of points and lines, polygons, which define boundary shapes, and spaces, which define the areas of interest in a space plan. Arvin and House (1999) more fully describes each of these elements, which are reviewed here to supply the concepts needed to understand how design objectives fit into the underlying system.

Nodes

A node is a point in space on which a force can be applied, and is the element on which all other physically based space planning elements are based. The type of a node determines how it will act when a force is applied to it. The two types of nodes we use are the point node and the line node.

Figure 3a shows a point node. A force *f* applied to a point node is not constrained in any way, and thus acceleration is in the direction of the applied force. We use point nodes to define the center of spaces. Figure 3b shows a line node. Line nodes are constrained such that they can move only in a direction perpendicular to the line it defines. Thus, only *f'*, the component of force *f* perpendicular to the line node (i.e. in the direction of normal vector *n*), acts to accelerate the line node. We use line nodes to define the polygonal edges of space boundaries.

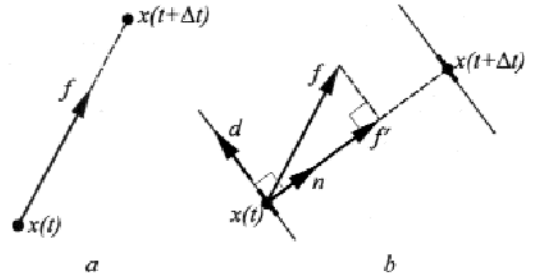

Figure 3. Node types.

Polygonal shapes

Shapes are limited to polygons with non-crossing edges and are defined as an ordered list of line nodes, each connected to a common center node. Figure 4 shows an arbitrary n-sided polygon, with center node *c*, edge line nodes *e*, and vertices *v*. Each vertex v_i can be found as the intersection of the direction vectors of e_i and e_{i+1} (where the operator + is addition modulo *n*). An ordered list of line nodes with constant direc-

tion vectors and varying positions allows for an infinite number of shapes, depending on the distance between the center and each line node. We attach springs between the center node and each line node, setting the spring rest lengths to specify the desired shape. The advantage of using a line node representation is that forces can be applied to the individual segments without having to worry about maintaining their orientation.

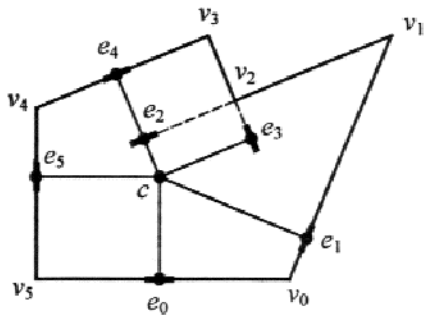

Figure 4. A polygonal shape with edge line nodes.

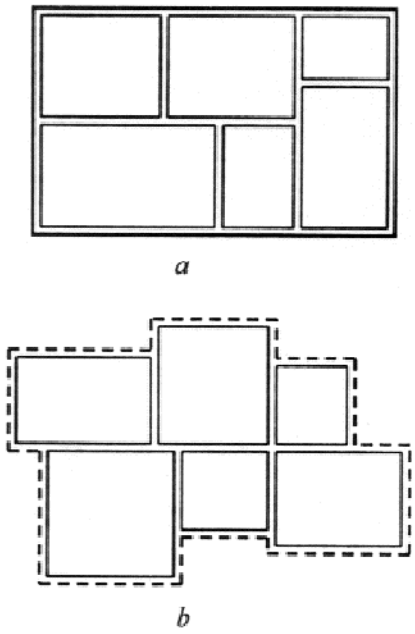

Figure 5. Parent space (a) with and (b) without a defined boundary.

Spaces

There is only one basic spatial unit, which is used to define any arbitrary area or volume. The data structure for a space contains a node, and a polygon, with the center of the polygon equal to the space node.

Any space may contain any number of child spaces, such that a hierarchy of spaces of arbitrary depth can be defined, similar to that described by Flemming and Chien (1995). This hierarchy can be represented as a directed acyclic graph. The set of nodes with a common parent node defines a separate dynamic simulation system. For example, a set of departments in an office building define one dynamic system, whose solution determines the relationships and dimensions of each department, and the set of rooms in a single department defines another dynamic system, whose solution determines the relationships and dimensions of each room. The definition of a parental boundary also determines how a system of child spaces resolves itself, as shown in Figure 5.

Design objectives

Given a set of spaces defined for a specific design problem, a set of design objectives determines the possible space plan solutions. As noted before, design objectives are separated into two categories, topological objectives and geometric objectives. Design objectives need to be defined in terms that are familiar to designers, they need to cause design elements to behave as expected, and they need to enable designs to emerge through a process of objective manipulation.

Topological Design Objectives

Topological design objectives influence the location of spaces relative to each other. Any objective that results in a force being applied to a space node is a topological objective. All topological objectives minimally specify a

space whose location the objective is trying to influence, and a vector, which specifies the direction and magnitude of the force being applied. The differences among the various topological objectives lie in the manner in which the direction vector is specified.

Adjacency Objective. An Adjacency Objective is a topological objective that influences how one space is positioned in relation to another. For example, two spaces that have a large amount of traffic between them may be specified with an immediate adjacency. The data structure for an adjacency objective contains two spaces, the strength of the adjacency relationship, and the resting distance between the centers of each space. The center nodes of each space are connected with a spring, which applies forces to the nodes depending on the distance between them. Using circular spaces as an example, the rest length of the spring may be the sum of the radii of each space. If the spaces are too far apart, the spring will produce forces on each space node that attempt to move them together, and vice versa. The spring constant defines the strength of the adjacency, and is initially set on a scale of 0.0 to 1.0 relative to other adjacency spring constants. Table 1 shows how descriptive terms for adjacency relationships might be mapped into numerical values for spring constants. Figure 6 shows two spaces connected with an adjacency objective.

Adjacency requirement	Spring constant
Immediate	1.0
Important	0.7
Convenient	0.3
Unimportant	0.0

Table 1. Adjacency Spring Constants.

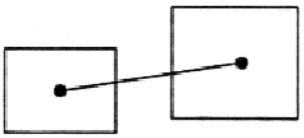

Figure 6. Adjacency Objective.

Orientation Objective. An orientation Objective is the simplest of the topological objectives and influences where a space is located on the building perimeter. An orientation objective can be used, for example, to specify a particular view for an office, or that a public parking area needs to be near the site entrance. The data structure for an orientation objective contains a space and a direction vector, which points to the side of the set of sibling spaces where the space needs to be located. The direction vector can be specified with an angle, a vector, or with descriptive terms such as Northeast or Southwest. A constant force is applied in the direction of the direction vector, such that the space tends to move in that direction. Figure 7 shows a Northeast orientation objective applied to a single space.

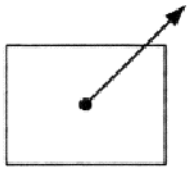

Figure 7. Orientation Objective.

Interior Objective. An Interior Objective is a topological objective that attempts to locate a space in the center of its set of sibling spaces. An interior objective may be used, for example, for spaces that have privacy or security requirements that may be better achieved by being located away from exterior walls. The data structure for an interior objective contains a space and a direction vector. The parental center of a space is defined as the average position of each of its sibling's positions. The direction vector starts from the space center node, and always points toward the parental center. A force is applied in the direction of the direction vector, thus tending to move the space toward the parental center, and away from the parental boundary. The magnitude of the force is proportional to the distance between the space center node and the parental center. Figure 8 shows a set of sibling spaces with parental center c. Two of the spaces have interior objectives spec-

ified, shown with direction vectors i_1 and i_2 pointing toward c.

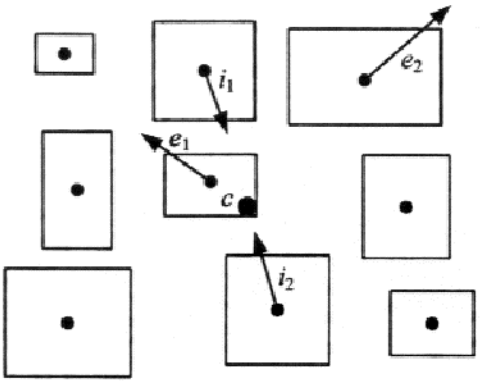

Figure 8. Interior and Exterior Objectives.

Exterior Objective. An Exterior Objective is the opposite of an interior objective, and is a topological objective that attempts to locate a space toward the boundary edges of its parental shape. An exterior objective may be used, for example, to specify that a space has daylighting requirements. The data structure for an exterior objective contains a space and a direction vector. The direction vector starts from the space center node, and always points away from the parental center. If the space center coincides with the parental center, the direction vector points toward the nearest edge of the bounding box surrounding the sibling centers. If the direction vector is still undefined due to equal distances to each bounding box edge, a random direction is chosen. A force is applied in the direction of the direction vector, thus tending to move the space away from the parental center and toward the parental boundary. The magnitude of the force is inversely proportional to the distance of the space from the parental center. Two of the spaces in Figure 8 have exterior objectives specified, shown with direction vectors e1 and e2 pointing away from c. A single space should not have an interior, exterior, and orientation objective specified at the same time.

Separation Objective. A Separation Objective is the opposite of an adjacency objective, and is a topological objective that attempts to locate one space away from another. A separation objective may be used, for example, to keep a private space and a public space on opposite sides of a building. Separation objectives are problematic, in that the force applied to the specified spaces needs to be large enough to move them apart, but not so large that the system they belong to separates into two different sets. A simple way to implement a separation objective is to set the rest length of the spring to a value larger than the sum of the radii of each space's circular representation, although this does not address the problem of spaces separating into multiple sets. We are currently looking into other ways to address this problem.

Geometric Design Objectives

Geometric design objectives influence the dimensions of space boundary edges. Any objective that results in a force being applied to an edge is a geometric objective. All geometric objectives minimally specify a node or set of nodes whose location the objective is trying to influence.

Alignment Objective. An Alignment Objective is a type of geometric objective that attempts to align two nodes. The data structure for an alignment objective contains two nodes; at least one of which must be a line node. If each node is a line node, the direction vectors of both need to be parallel. Each node is connected with a spring with a rest length of zero. Figure 9 shows the effect of an alignment objective on two parallel line nodes.

Any non-trivial design problem will probably be over-defined, in that there may be more than one objective applied to the same node. In such a condition, an alignment objective represented with a simple spring and dashpot will rarely align its two nodes. For this reason we use an integral spring. The force applied by an integral spring contains the spring and dashpot components described earlier, as well as a third component that continuously increases until the desired rest length is achieved. This third com-

ponent is the sum of the errors of previous time steps, where the error at a given time is the difference between the length at that time and the rest length. As the simulation time increases, this value increases until the error is zero, causing the spring to apply just enough force to make the current length equal to the rest length.

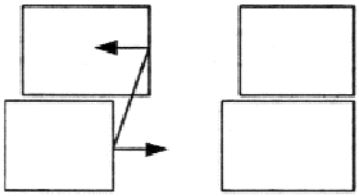

Figure 9. Alignment Objective.

Offset Objective. An Offset Objective is a generalized Alignment Objective. As opposed to an alignment objective where the rest length of the spring is set to zero, for an offset objective the user specifies a non-zero rest length. Figure 10 shows the effect of an offset objective on two parallel line nodes.

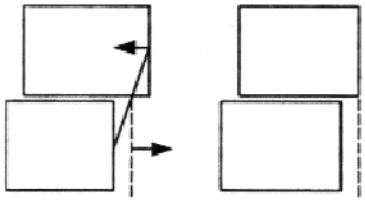

Figure 10. Offset Objective.

Area Objective. An Area Objective is a type of geometric objective that attempts to maintain the area of a polygonal shape. The data structure for an area objective contains a polygon, an area, and an area range. The edge springs of the polygon indirectly maintain the correct area of the boundary polygon, but, in the case of a rectangle, if the springs along one dimension are shortened the springs along the other are not lengthened accordingly. When an area objective is applied, if the springs along one dimension are shortened, thus decreasing the area of the polygon, the area objective applies additional forces to all edge nodes to attempt to increase its area. If the actual area is within the bounds of the area range, no forces are applied, otherwise a force is applied with magnitude proportional to the difference between the actual area and the nearer bound of the area range. Figure 11 shows a rectangular polygon, with the desired area shown as a dashed rectangle, the area range shown as dotted rectangles, and area objective forces applied to the edge line nodes.

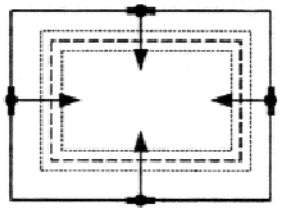

Figure 11. Area Objective.

Proportion Objective. A Proportional Objective is a type of geometric objective that attempts to maintain the specified proportions of a polygonal shape. It is similar in concept and application to the area objective. If the proportions of a polygon deviate beyond the range specified, forces are applied to the polygonal edge nodes to attempt to bring them back into range. Figure 12 shows a rectangular polygon, with the desired proportion shown as a dashed rectangle, the proportion range shown as dotted rectangles, and proportion objective forces applied to the edge line nodes.

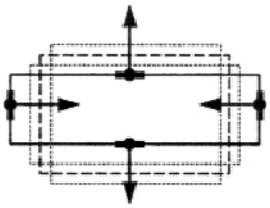

Figure 12. Proportion Objective.

Gravity Objective. A Gravity Objective is a type of geometric objective that attempts to remove gaps between adjacent spaces by making them gravitate toward each other. A global gravity objective continuously checks for gaps between adjacent spaces, and then applies an offset objective between them, with the offset set at the user specified interior wall thickness. Local gravity objectives can also be applied during the design process by manually specifying an offset objective between two spaces, or by simply moving one space until it comes in resting contact with another. In our system, resting contacts are implemented with dynamic constraints, as described in Arvin and House (1999), and have the effect of "gluing" spaces together. Figure 13a shows a simple system of spaces with only adjacency objectives being applied. Figure 13b shows the result of applying a gravity objective to the same system.

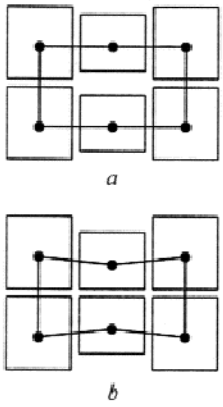

Figure 13. Gravity Objective.

Dynamic simulation

Once a set of spaces and objectives has been defined, a dynamic simulation runs through a series of phases to produce a layout solution. First, topological relationships between spaces are resolved. Second, the geometric position of walls separating the spaces is resolved. Finally, the designer interacts with the design by modifying the set of design objectives, thereby modifying the design itself. This is conceptually a linear process, but in reality it is highly circular. Every modification of design objectives causes the simulation to loop to either topological resolution or geometric resolution, depending on the type of design objective being modified.

Topological Resolution

The first phase in solving a space layout is to determine the location of each space relative to all other spaces. In this phase only topological objectives are applied. For collision detection, boundary shapes are treated as circles so spaces are able to slide around each other. If polygonal boundary shapes were used, corners might catch on each other and keep one space from being able to move to the other side of another. The dynamic simulation runs until the system is in equilibrium, which is defined as the point in time when the magnitude of all velocities are below a small threshold.

Geometric Resolution

Once the topological simulation has reached equilibrium, the second phase is started, during which geometric objectives are applied and topological objectives are turned off. In this phase space boundaries are switched from a circular to a polygonal representation. Collision detection and response then act to keep spaces from overlapping, resulting in an arrangement that is very close to a recognizable building floor plan. We use constrained dynamics to maintain the separation between spaces, which are more fully described in Arvin and House (1999) and Witkin and Baraff (1997).

Design Interaction

Once a geometric simulation has reached equilibrium, the designer can begin to analyze and interact with the design by modifying existing objectives and adding new ones. Here is where the true power of this approach is realized. The designer interactively manipulates the design via objectives rather than via geometry, allowing him or her to concentrate on the design itself rather than on the mechanics of geometric transformations.

Results

Figure 14 shows two results from a sample design problem using some of the design objectives we have implemented as of this writing. Figures 14a and 14b show the same set of spaces with the same initial random positions, but with a different set of topological objectives. Adjacency objectives are solid lines connecting spaces, the width of the line representing the strength of the adjacency. Separation objectives are dashed lines connecting spaces, in this example all with the same separation strength. Figure 14b shows the same problem as in 14a, but with the addition of an interior objective for space 3, an exterior objective for space 6, and an orientation objective for space 7.

Figures 14c through 14g show the process of resolving a space plan from the initial state shown in 14b. 14c shows the result of topological resolution after the simulation has reached equilibrium. Recall that during topological resolution space boundaries are treated as circles for collision detection purposes. Note the location of space 9, which is connected to all other spaces with separation objectives. This physical separation demonstrates the problem of separation objectives described above. 14d shows the transition of space boundary representation from circles to polygonal shapes, and shows the many gaps and overlaps between shapes. 14e shows the first step in geometric resolution, with all topological objectives turned off, and geometric objectives except gravity turned on. The overlaps are removed, but the gaps remain. 14f shows the result of adding manual gravity objectives, which removes the gaps. This figure also shows the beginnings of manual design interaction with the minor relocation of spaces 4, 7, and 9. Finally, 14g shows more designer manipulation with the addition of alignment objectives to clean up the outer walls.

Figure 14h shows one step near the end of the process of resolving a space plan from the other initial state shown in 14a. It is at a similar state in the process as Figure 14f, but without any manual space relocations. Recall that 14a and 14b have the same set of adjacency objectives. Note how the addition of interior, exterior, and direction objectives on spaces 3, 6, and 7, respectively, affect their final locations in Figure 14f.

Discussion

We feel that our prototype system provides a convincing demonstration of the use of physically based design objectives and the attractiveness of the responsive design approach. It is difficult to convey, in a paper, the experience of working with our system. But, it really does evoke the feeling that one is working with a "living" design, one that responds to the user in ways consistent with programmatic objectives while still providing a high degree of intuitive designer control. The fact that the physically based objectives do most of the work allows one to explore many design possibilities easily, released from the need to be continually aware of spatial areas and primitive geometric elements.

Within our prototype system, the quality of interaction when manipulating spaces and objectives is, for the most part, as we envisioned it to be. When editing a space plan, the designer is able to move spaces around easily to try out new designs quickly. The addition of alignment objectives to our repertoire of design objectives was a very exciting moment. They revealed the potential of this approach to be a useful design tool as opposed to just another method for producing automated space layouts.

We were initially surprised at the few types of design objectives that came easily to mind. Part of this is due to the limited number of geometric elements we need to deal with, and part is due to the definition of the problem. The set of topological objectives is limited to the number of ways a vector can be applied to a point, i.e. the number of ways a force can be applied to the center of a space. Similarly, the set of geometric objectives is limited to the number of ways a force can be applied to the line node representing a polygonal edge. There are only so many ways a vector can relate to a point, which limits the number of possible design objectives. Although this may seem a shortcoming of our overall approach, within the domain of space layout planning the few objectives we have dis-

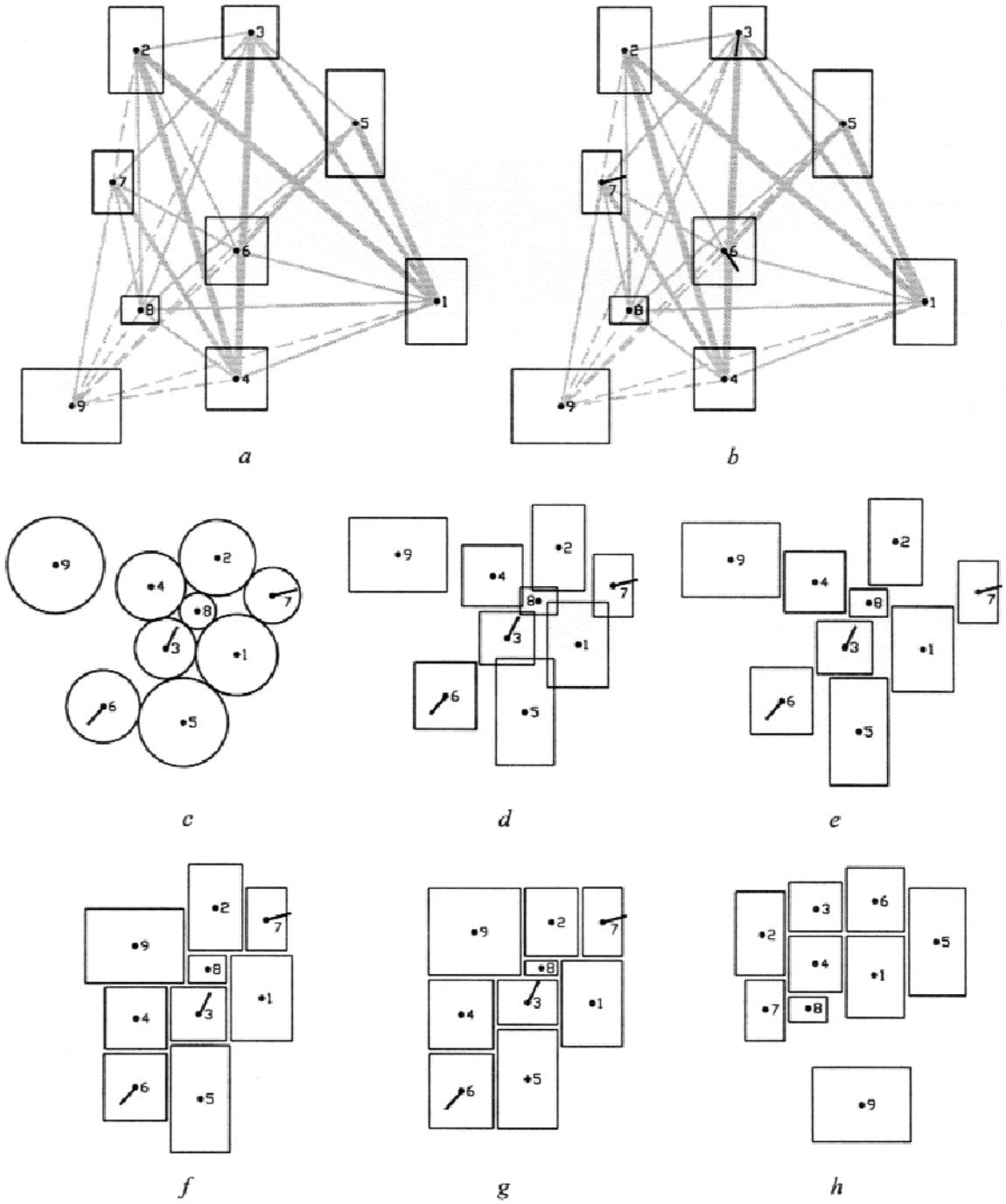

Figure 14. Sample Results.

cussed enable the designer to produce a wide variety of results. It remains to be seen if these objectives are adequate as applied to the overall design process.

Future work

Our physically based approach to modeling design objectives is a previously unexplored con-cept, and raises many new questions and presents many opportunities for future work. Some of these are obvious and require answers and elaboration in order to make this approach truly useful to space planners. Other questions are of a more fundamental nature.

In the immediate short term, we plan to explore:

• What other types of design intentions can be translated into design objectives? Do the objectives we propose cover most of the needs of architectural designers?

• Will the objectives described here be effective when this approach is modified to handle circulation? Or will modifications and additional objectives be required?

• Will the objectives described here be effective when this approach is extended to three dimensions?

• What is an effective way of displaying the various forces on a space so that they are meaning-ful to designers and allow them to understand the conflicting nature of a set of design objectives?

To validate and more firmly ground our work we plan to explore the following fundamental questions:

• How does this objective-based approach fit into the architectural design process?

• Does our use of physically based design objectives truly provide a responsive design process?

• How much of the underlying physical model should be revealed to designers? Do they need to know about the underlying forces and the details of the physically based system, or is it better for them to understand how it works through the use of another metaphor? Will it be possible for designers to extend the repertoire of design objectives without having a thorough knowledge of the underlying system?

Conclusion

We have described how a variety of design objectives can be modeled as forces applied to physically based space planning elements. We have also demonstrated, with a simple prototype sys-tem, how these objectives can be used in a space planning design process. Our preliminary results with the prototype indicate that a powerful set of manipulable design objectives is essential to providing a responsive design process, and that these tools integrate well with the exploratory nature of the design development process and can be enjoyable to use.

Acknowledgements

This work was supported in part by an O. N. Mitchell, Jr. Endowed Fellowship in Construction Management, the Visualization Laboratory and the Department of Architecture in the College of Architecture Texas A&M University.

References

Arvin, S. A., and House, D. H. (1999). Making Designs Come Alive: Using Physically Based Modeling Techniques in Space Layout Planning. G. Augenbroe and C. Eastman (Eds.), *Computers in Building: Proceedings of the CAADfutures '99 Conference* (pp. 245-262), Atlanta, GA: Kluwer Academic Publishers.

Balachandran, M., and Gero, J. S. (1987). Dimensioning of architectural floor plans under conflicting objectives. *Environment and Planning B* (14), 29-37.

Baraff, D. (1989). Analytical methods for dynamic simulation of non-penetrating rigid bodies. *Proceedings of SIGGRAPH '89* (pp. 223-232), Boston, MA: ACM SIGGRAPH.

Barzel, R., and Barr, A. H. (1988). A Modeling System Based on Dynamic Constraints. *Proceedings of SIGGRAPH '88* (pp. 179-188), Atlanta, GA: ACM SIGGRAPH.

Flemming, U. (1987). The role of shape grammars in the analysis and creation of designs. Y. E. Kalay (Ed.). *Computability of Design* (pp. 245-272). New York, NY: John Wiley & Sons.

Flemming, U. (1989). More on the representation and generation of loosely packed arrangements of rectangles. *Environment and Planning B* (16), 327-359.

Flemming, U., and Chien, S. F. (1995). Schematic Layout Design in SEED Environment. *Journal of Architectural Engineering* (1:4), 162-169.

Flemming, U., and Woodbury, R. (1995). Software Environment to Support Early Phases in Building Design (SEED): Overview. *Journal of Architectural Engineering* (1:4), 147-152.

Gero, J. S., and Kazakov, V. A. (1998). Evolving design genes in space layout planning problems. *Artificial Intelligence in Engineering* (12:3), 163-176.

Gross, M., Ervin, S., Anderson, J., and Fleisher, A. (1986). Designing with constraints. Y. E. Kalay (Ed.). *Computability of Design* (pp. 53-68). New York, NY: John Wiley & Sons.

Harada, M. (1997). Discrete/Continuous Design Exploration by Direct Manipulation. Ph.D. Dissertation. Carnegie Mellon University.

Harada, M., Witkin, A., and Baraff, D. (1995). Interactive Physically-Based Manipulation of Discrete/Continuous Models. *Proceedings of SIGGRAPH '95* (pp. 199-208), Los Angeles, CA: ACM SIGGRAPH.

Jo, J. H., and Gero, J. S. (1998). Space layout planning using an evolutionary approach. *Artificial Intelligence in Engineering* (12:3), 149-162.

Liggett, R. S., and Mitchell, W. J. (1981). Optimal space planning in practice. *Computer-Aided Design* (13:5), 277-288.

Mandal, C., Qin, H, and Vemuri, B. C. (1997). Dynamic Smooth Subdivision Surfaces for Data Visualization. *Proceedings of IEEE Visualization '97* (pp. 371-377), Phoenix, AZ.

Martini, K. (1995). Hierarchical geometric constraints for building design. *Computer-Aided Design* (27:3), 181-191.

Mitchell, W. J. (1977). *Computer-Aided Architectural Design*. New York, NY: Petrocelli.

Pfefferkorn, C. E. (1975). The Design Problem Solver: A System for Designing Equipment and Furniture Layouts. C. M. Eastman (Ed.). *Spatial Synthesis in Computer-Aided Building Design* (pp. 98-146). New York, NY: John Wiley & Sons.

Qin, H., Mandal, C., and Vemuri, B. C. (1998). Dynamic Catmull-Clark Subdivision Surfaces. *IEEE Transactions on Visualization and Computer Graphics* (4:3), 215-229.

Simon, H. A. (1973). The structure of ill-structured problems. *Artificial Intelligence* (4), 181-201.

Terzopoulos, D., Platt, J., Barr, A., and Fleischer, K. (1987). Elastically deformable models. *Proceedings of SIGGRAPH '87* (pp. 205-214), Anaheim, CA: ACM SIGGRAPH.

Tobin, K. L. (1991). Constrain-Based Three-Dimensional Modeling as a Design Tool. G. Goldman and M. Zdepski (Eds.), *Reality and Virtual Reality, 1991 ACADIA Proceedings* (pp. 193-209), University of California, Los Angeles.

Weinzapfel, G., and Handel, S. (1975). IMAGE: Computer Assistant for Architectural Design. C. M. Eastman (Ed.). *Spatial Synthesis in Computer-Aided Building Design* (pp. 61-97). New York, NY: John Wiley & Sons.

Witkin, A., and Baraff, D. (1997). *Physically Based Modeling: Principles and Practice. SIGGRAPH '97* Course Notes, Course 19, ACM SIGGRAPH.

Witkin, A., and Kass, M. (1989). Spacetime constraints. *Proceedings of SIGGRAPH '88* (pp. 159-168), Atlanta, GA: ACM SIGGRAPH.

Yoon, K. B. (1992). *A Constraint Model of Space Planning*. Southampton, UK: Computational Mechanics Publications.