# TRANSFORMATIONAL DESIGN

**Costas Terzidis**

Graduate Program in Computer Aided Architectural Design
Department of Architecture
The Ohio State University

**ABSTRACT**

The use of dynamically executable transformations and their orchestration in time is discussed and explored as a design tool. The aim has been to accommodate the dynamic character of architectural design during its form searching stages. The transformation and reformation of architectural elements is executed in real time under the direction of a user, who, by controlling the rhythm and the speed, orchestrates the compositional evolution of an architectural parti.

## 1. INTRODUCTION

An almost unique characteristic of Architecture is that it is both dynamic and static. It is dynamic when viewed as the design process which has its roots in historical precedents of culture and the arts and which manipulates entities which are typically of an elastic character. It becomes static when it has to freeze at a certain state so that it may be built. In other words, architecture is static when viewed through individual buildings. It is dynamic when these buildings are viewed as instances of a continuum which derives from the past and projects into the future.

The static nature of architecture is currently served reasonably well by the available commercial CAD systems. However, whether it is for the drafting of 2-D construction drawings or for the production of 3-D renderings, these systems can only be used after an architectural design has been more or less finalized. To the best of my knowledge there is no CAD system which is capable of dealing with the architectural form while it is still soft and in transition.

Imagine a computer screen where all is in motion. A variety of elements, either of a pure form, such as the cube, or of a symbolic form signifying a memory of the past, are paraded and can be selected and/or operated by a user who enters commands through the mouse. As organizational structures are introduced, they are executed in a continuous mode and in a way such that all the in-between stages are transparent and become part of the compositional experience. Processes may be reversed individually or in groups, may be applied at different speeds, or they may be interrupted and redirected. The morphological evolution of centuries may thus be replayed with accuracy or may be extrapolated into directions which were missed by history. For example, a doric temple may be crossprogrammed with the Barcelona Pavillion, a path for which there is no known historical precedent.

When wire frame representations are used for solids, the computer power to execute multiple transformations of forms is today available even on the personal computers. What have yet to be fully explored are the necessary interfaces and their correspondence to the architectural semantics. The work presented here discusses an experiment in that direction. It was undertaken as a master level thesis project.

The key theme is the transformation of physical entities. When these entities are viewed as parts of an architectural parti, then the theme becomes the transformation of architectural forms. The transformations explored go beyond the simple symmetry transformations of translation, rotation and scale. They include the transformation of one shape to another. These concepts and their implementation in an experimental CAD system are discussed by the remainder of this paper, which concludes with illustrations of the potential of the system as a design tool.

## 2. INFLUENCES

Transformations of forms have been used rather extensively in the classic Greek architecture, as well as in other periods, but they more dramatically manifest themselves in the work of contemporary designers such as Peter Eisenman. Eisenman [1986] regards the idea of transformation as a process of change between architectural types. He considers the history of Architecture as a dynamic continuum of type transformations: "*..Transformation, while it did not necessarily suggest any ideal order, presumed that the significance of the final form resided, in part, in the process itself; in the capacity of the object to reveal its own origins and processes, to register back to an original type, by a kind of reverse mental process. It was hermetic and internalized."*

In the work of Daniel Libeskind, architectural elements such as beams and columns are placed in space in a way such that they appear as if they are *moving*. It seems that there are some relationships which determine the position of each object in such a way that the moving image can better be described as *a dancing image* (Libeskind [1981]).

As forms are transformed, they may be combined in accordance to some algorithmic logic. In Bernard Tschumi's terms, forms may be disprogrammed,or crossprogrammed or transprogrammed, through some combinative method, may be even act of creation. *" ... all 'new' architecture implies the idea of combination, ... all form is the result of a combination. ... architecture is … a complex process of transformational relations."* (Tschumi [1983]).

An implementation of transformational processes in a CAAD system has been reported by Yessios [1987]. The system was used in design exercises which were recorded on video tapes to fully reflect the dynamic essence of the explorations. Yessios's algorithms dealt with 2-D shapes. The work presented here, in essence, is extending these algorithms into the third dimension and attempts to generalize their semantics.

## 3. THE IMPLEMENTATION

*A* ***transformation***, in this paper, is a process in which the form of an object changes gradually in order to obtain another form. The term "transformation" has been traditionally used to mean the symmetry operations of translation, rotation and scaling. These operations are typically applied uniformly to the geometry of all the points of an object. The operations discussed in this paper include the symmetry operations. However, the geometries of individual points are transformed independently of the other points in an object and the transformations are dictated by processes rather than by the application of transformation formulas.

In addition to the geometry, the topology of an object may also be altered. This is necessary when the object to which the form of another object changes has a different topology. To avoid confusion with the traditional use of the term "transformation", the operations discussed here will be called ***reformations***. The operation of reformation basically consists of the selection of two objects and the assignment of a value to *n*, the number of in-between steps. The first object then transforms into the second in *n* steps. This process is illustrated in Figure 1.

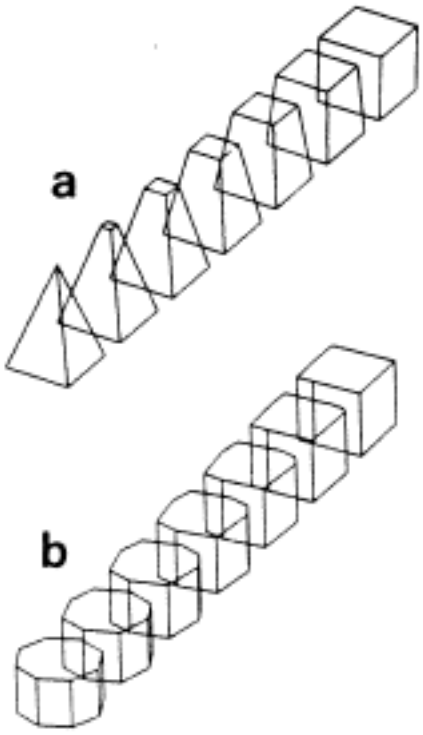


Figure 1. Reformations. (a) Prism to cube. (b) Cylinder (aprox.) to cube.

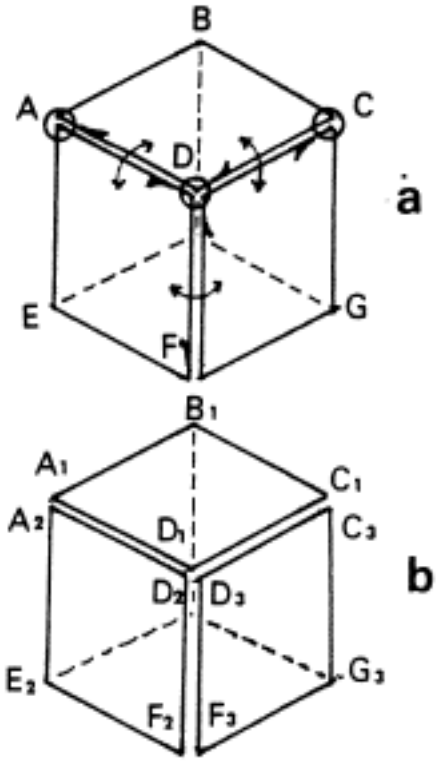


Figure 2. The connectivity constraints: (a) applies; (b) relaxed.

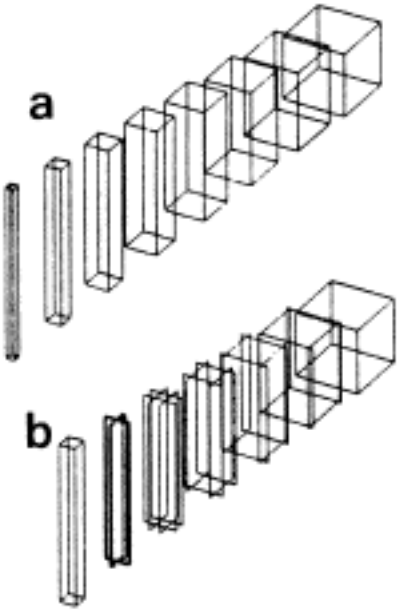


Figure 3. Reformations. (a) Volume to volume with connectivity constraint; (b) face to face with no connectivity constraint.

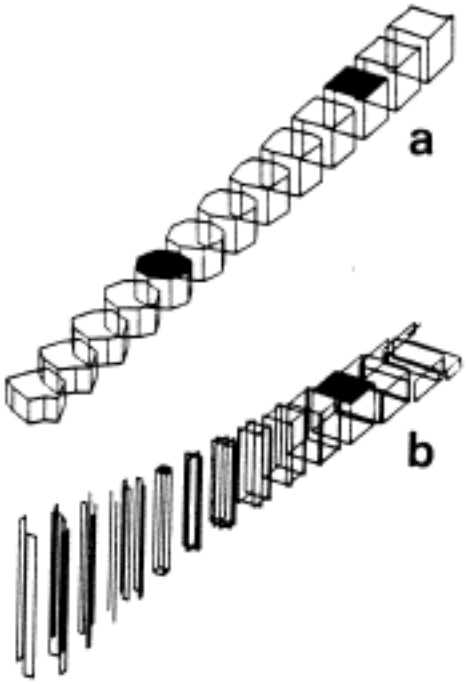


Figure 4. Reformations extrapolated beyond their terminal forms.

Reformations can be applied at any of the topological levels of solid geometry, namely, points, edges, faces and volumes. They can be applied by either preserving or relaxing some of the connectivity attributes of solid geometry. These attributes are reviewed in Figure 2. The cube shown in 2(a) satisfies two connectivity conditions. Points such as D are shared by three faces. If D is moved, it will effect all the faces which share it. Similarly, all edges are tied to a reversely coincident segment of another face, and when they are transformed (moved or scaled or rotated), they effect both of the faces which share them/.

In contrast, these connectivity conditions have been relaxed in 2(b). Independent copies of D (namely, D1, D2 and D3) are used for each of the three faces which share the same geometric value of D. Thus, when one of the D's, say D1, is moved or rotated, it effects the form of only one face. Similarly with the edges, whenever their connections to their reversely coincident edges are relaxed.

It should be self-evident that, depending upon the topological level and the connectivity constraints applied, reformations produce different results. The implementation presented here has experimented with two types of reformations: (a) ***volume to volume with connectivity constraints***, and (b) ***face to face with no connectivity constraints***. The two cases are illustrated in Figure 3. It should be noted that the connectivity attribute here refers to the connections implied when points and edges are shared by faces. It does not refer to the topological connections which represent the vector lines which delineate faces.

From a practical point of view, when the connectivity constraint is applied, the reformation preserves the structural integrity of the objects involved. That is, the form of an object changes to the form of another object as a single entity (Figure 3(a)). When the connectivity constraint is relaxed, during the reformation process, the form of an object is dissolved, but the object reconstructs itself at the end (Figure 3(b)). Both variations are useful for transforming architectural forms.

***Orchestration*** is a term used to describe the actions of selecting, assigning, directing and evaluating the performance of objects which participate in a reformation. Reformations can happen concurrently and/or at different speeds. The result is a ***moving image*** the behavior of which becomes the responsibility of the user. As in an orchestra performance, the architect/composer selects a number of objects he/she wants to include, assigns the proper transformation paths and speeds, and then directs the performance through time, form and color.

The essence of such transformational design is not that much in the final form but rather in the intermediate phases these reformations pass through, as well as, in the extrapolations which go beyond the final form. The latter is a significant by-product of the reformation operations. Having established the process which transforms one form to another, in addition to the in-between forms which the process can produce, it can also be applied to produce forms which are beyond the forms of the two original objects. Examples are shown in Figure 4, where (a) illustrates a volume to volume and (b) a face to face reformation. Both extrapolate beyond their end forms. Extrapolations beyond the terminal forms frequently produce results which can pleasantly surprise a designer.

The user has the capability, through the system, to modify and control the flow of the compositional evolution and replay it many times by varying some or all of the transformational parameters. The user controls the flow of the moving image through the mouse which can be moved in either one of two directions: horizontal (right-left) and vertical (up-down). The horizontal direction represents ***time*** while the vertical represents ***space.*** As the cursor is moved upwards or downwards the objects are transformed forwards or backwards, respectively. The speed is relative to the movement of the mouse. Similarly, when the user moves the cursor right or left, the whole scene rotates in space clockwise or counter-clockwise.

## 4. THE REFORMATION ALGORITHM

When an object is reformed, it changes its shape gradually in order to match the shape of the object it transforms to. A cube, for example, may be gradually transformed into a pyramid. From the user's point of view, there are always two objects: ***the initial object***, to which a reformation is applied, and the ***destination object*** , which is the object we have at the final step of the reformation. However, internally, there is only one object, which is transformed from one state (initial) into another (destination). This object combines characteristics of both of the objects which are involved in the reformation and will be called a ***hybrid object.*** In the general case, this object is actually composed of the topology of one object and the geometry of another.

Let O1->O2 be a transformation between two objects O1 and O2. The algorithm to create a hybrid object is as follows:

1) Compare O1 and O2. Mark as Omax the object with the most coordinates. Mark Omin the other.

2) Consider an xyz-axis system which passes through the centroid of Omax and divides it in eight quarter-blocks. Mark each vertex of Omax with a number corresponding to the quarter-block to which it belongs. This number is called the *orientation number*.

3) Do the same for Omin's vertices.

4) Compare each point of Omax and Omin. If the orientation numbers match, substitute the Omax vertex with the corresponding Omin vertex.

5) Interpolate the remaining (unmatched) points of Omax, so that they fall between the previous and the next vertex (found in step 4).

This algorithm constructs a hybrid object which has the topology of Omax, but its vertices match the vertices of Omin. This object appears like Omin, but topologically it is Omax.

After constructing the hybrid object, the increment by which this object will be changed in order to move from one form (initial) to the other (destination) needs to be found. Since we know in advance the initial and the destination object, we subtract the original object from the hybrid object, or the hybrid from the destination, depending on the direction of transformation and divide each by the number of steps the user has indicated. Those increments are stored and are progressively added to the hybrid object, which transforms from one state to another. This is a linear reformation, where a point moves along the line which connects the initial with the destination position. More elaborate paths can also be defined through mathematical functions, which may be applied uniformly to all points or different reformation paths may be assigned to different points of the hybrid object.

## 5. THE PERFORMANCE

A sequence of transformational events is called ***performance.*** The user orchestrates a performance by selecting objects, assigning tasks and setting them to run under his/her control. Depending on the way the tasks are set, the user can produce rhythm, synchronization, or pause. As illustrated in Figure 5, a transformational process can be any of the following types:

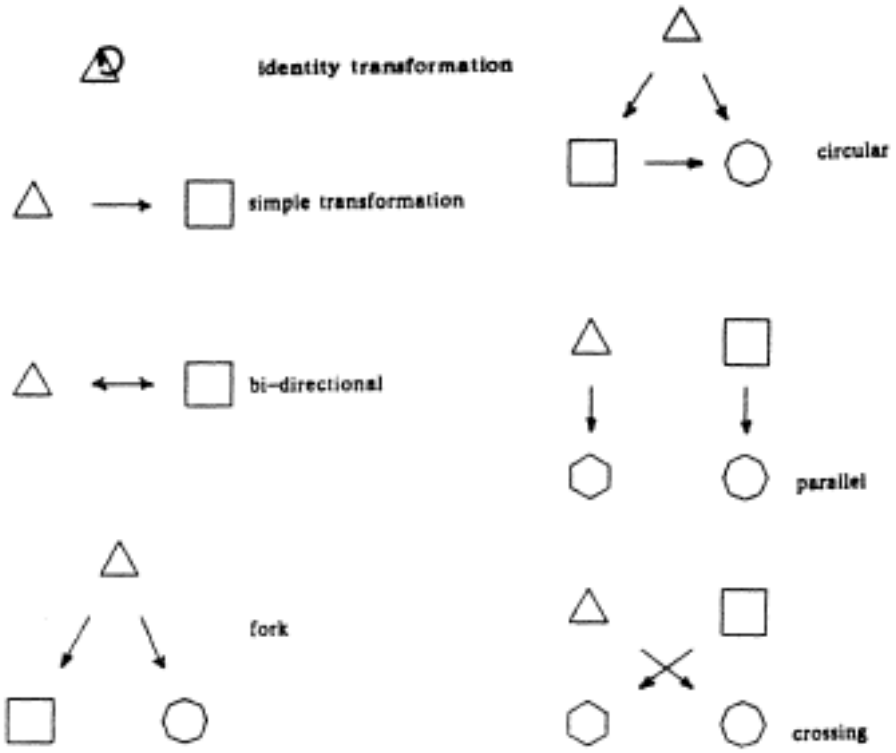


**Figure 5. Variations in which reformations can be applied.**

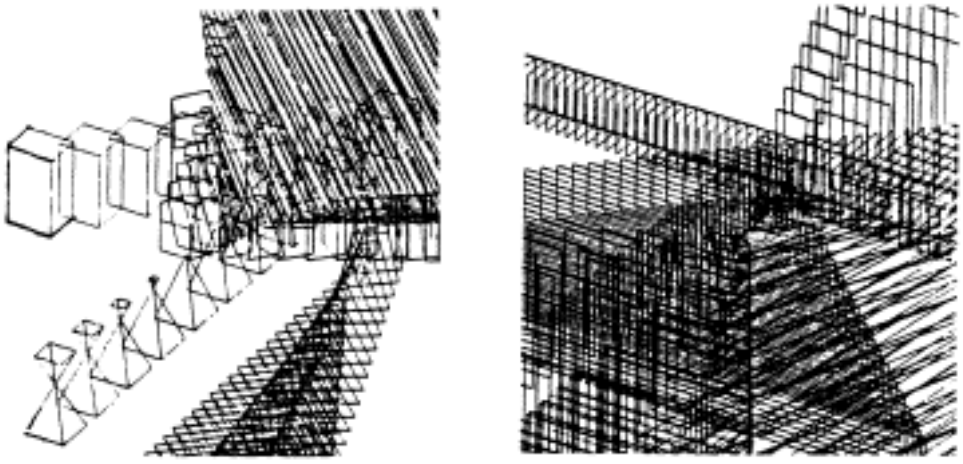

**Figure 6. Reformations leaving traces.**

- Transform an object into itself (identity transformation).

- Transform object 1 into object 2 and object 2 into object 1 (bi-directional).

- Transform object 1 into object 2 and object 1 into object 3 (fork).

- Transform object 1 into object 2, object 2 into object 3 and object 3 into object 1 (circular).

- Transform object 1 into object 3 and object 2 into object 4 in a parallel direction (parallel).

- Transform object 1 into object 3 and object 2 into object 4 crossing each other (cross).

In mathematical terms, a transformation can be viewed as a mapping function between one, two or more collections of points. Thus, it can be symbolically represented as $F(x,y,z)=(f(x,y,z),g(x,y,z),h(x,y,z))$. This representation is useful for implementation purposes, but this paper will not be concerned with further technical details.

Reformations can be executed in a continuous fashion or one at a time. As shown in Figure 6, their execution can leave traces of all the in-between stages or a single form can be shown at a time. In all these modes, the user does not interact, but the system controls the flow of the reformations. These modes have proven useful for previewing individual reformations. Once the desirable set of reformations and their parameters have been established, then they can be run under a user's interactive control, where the user plays the role of a conductor.

## 6. APPLICATIONS

The implementation of the reformations was intended to model design procedures which are actually used by notable designers. Whether we can all agree what crossprogramming or transprogramming exactly is, is probably insignificant. In general terms such design procedures borrow forms from the past or the contemporary, manipulate them through some algorithmic logic and may or may not mix them. Exactly how it is done and what the ultimate compositional goal is depends upon the individual designer and his/her stylistic preferences. Consequently, the implementation of the reformations were not intended to offer a

"recipe" for design, but rather a "tool" which individuals can use in their own ways.

As implemented, reformations can be used in a variety of ways and can be mixed and interfaced with other more traditional operations. Once the primitives of a design have been established, they are set in motion and perform reformations individually or in groups. The user directs these reformations and visually experiences the forms which are continuously generated and their interaction with each other. At any time, the process can be paused, replayed or reversed, thus giving the user an opportunity to select a certain manifestation of a form or a group of forms for inclusion in a design. These "saved" instances of forms can then be further acted upon by more conventional operations such as unions, intersections and differences or topological editing.

The examples illustrated in this section are intended to show some of the flavor involved with the use of the reformations. They are not intended to illustrate complete designs, which would actually be contrary to the spirit of the work presented. It should also be noted that the main virtue of the reformations is that they are dynamic and are executed in real time. Much of this flavor is certainly lost when presenting static images printed on paper.

The example in Figure 7 shows the application of reformations on distinct elements of Mies van der Rohe's Barcelona Pavillion (1929). Volume to volume reformations with a connectivity constraint are applied. The pairings and the directions of the reformations are shown in 7(a). The remaining illustrations are frozen instances taken out of the reformation continuum.

Figure 8 illustrates face to face reformations with no connectivity constraint. The reformations are defined over the faces of the nine-cube composition shown in 8(a). The pairings and the destinations are shown in 8(b). One in-between state of the reformation is shown in 8(c). As expected, these reformations preserve the integrity of their forms at the face level only. At the volume level, the form of the object is dissolved. Figure 9 illustrates a variation of the reformations in Figure 8. Only the faces which are actually transformed are shown. Those whose shapes remain unaltered have been filtered out. This negates the issue of whether or not the integrity of the forms is preserved at the volume level, since the filtering procedure has already reduced the solids to planar (faces only) compositions.

Figure 10 illustrates the use of reformations for crossprogramming the forms of two buildings: a Greek Doric Temple and Mies van der Rohe's Brick House. The definition of the reformation is shown in 10(b). The "crossing" of the two forms is

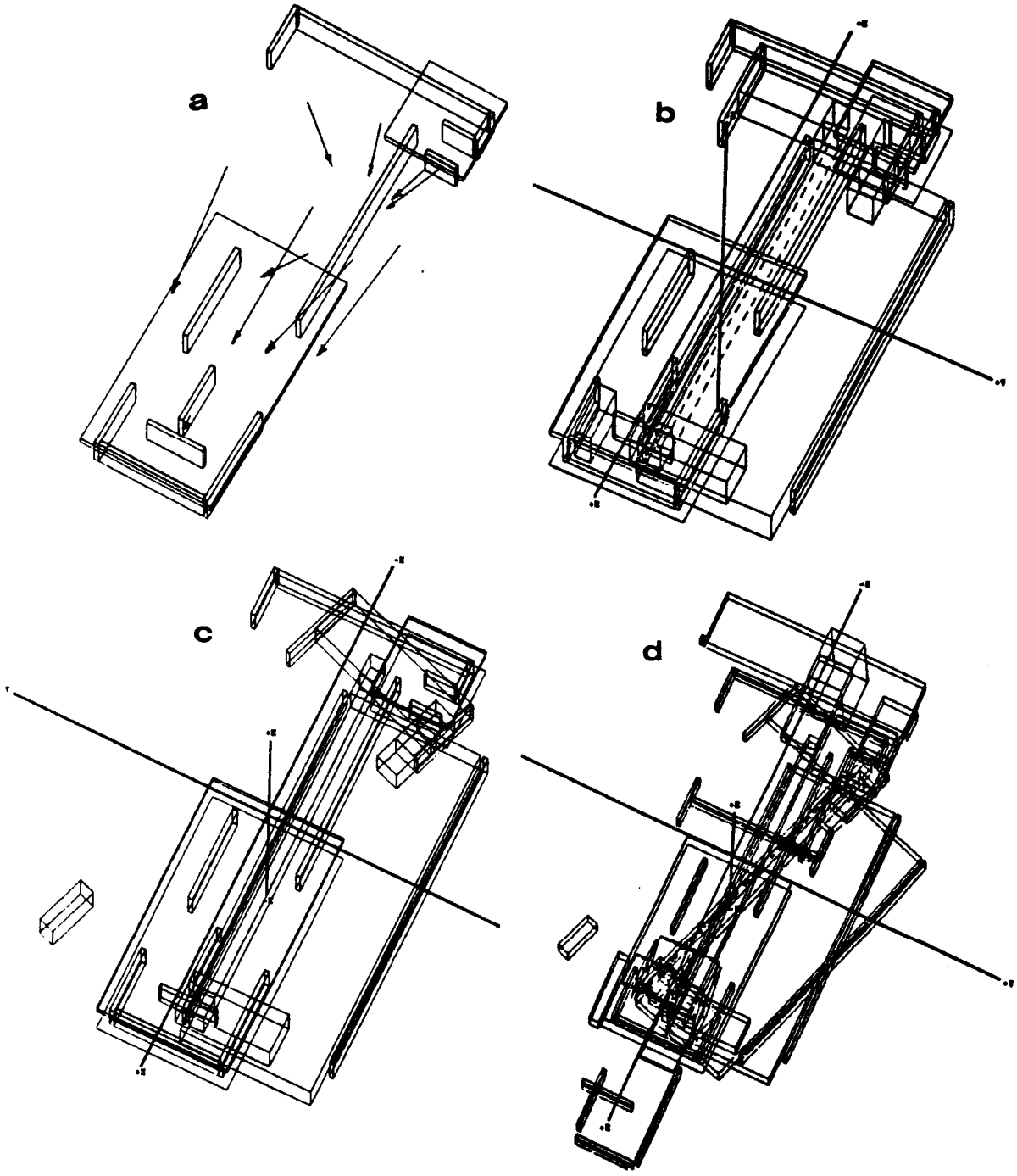


**Figure 7. Reformations applied in-between components of Mies van der Rohe's Barcelona Pavillion. (a) The pairings and the directions of the reformations. (b) - (c) Selected frozen instances of the reformation continuum.**

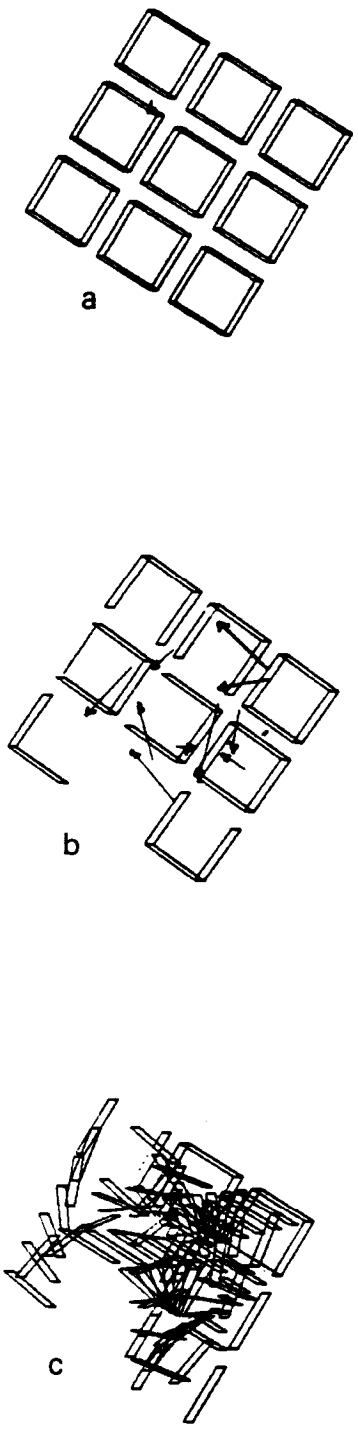


Figure 8. Face to face reformations applied on (a) a nine-cube composition. (b) Pairings and directions of reformations. (c) An in-between instance.

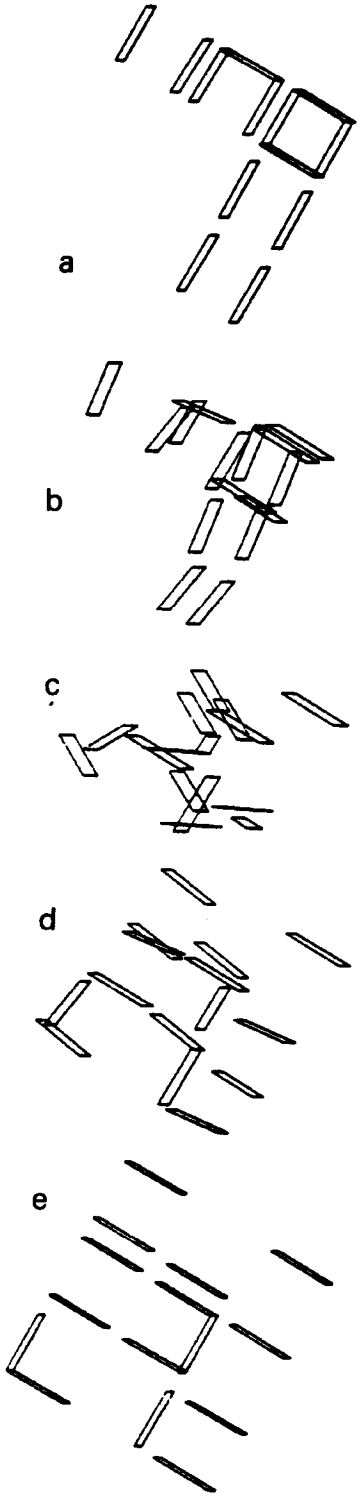


Figure 9. A filtered version of of the reformations in 8. (a) Initial form; (b) - (d) in-between states; (e) destination form.

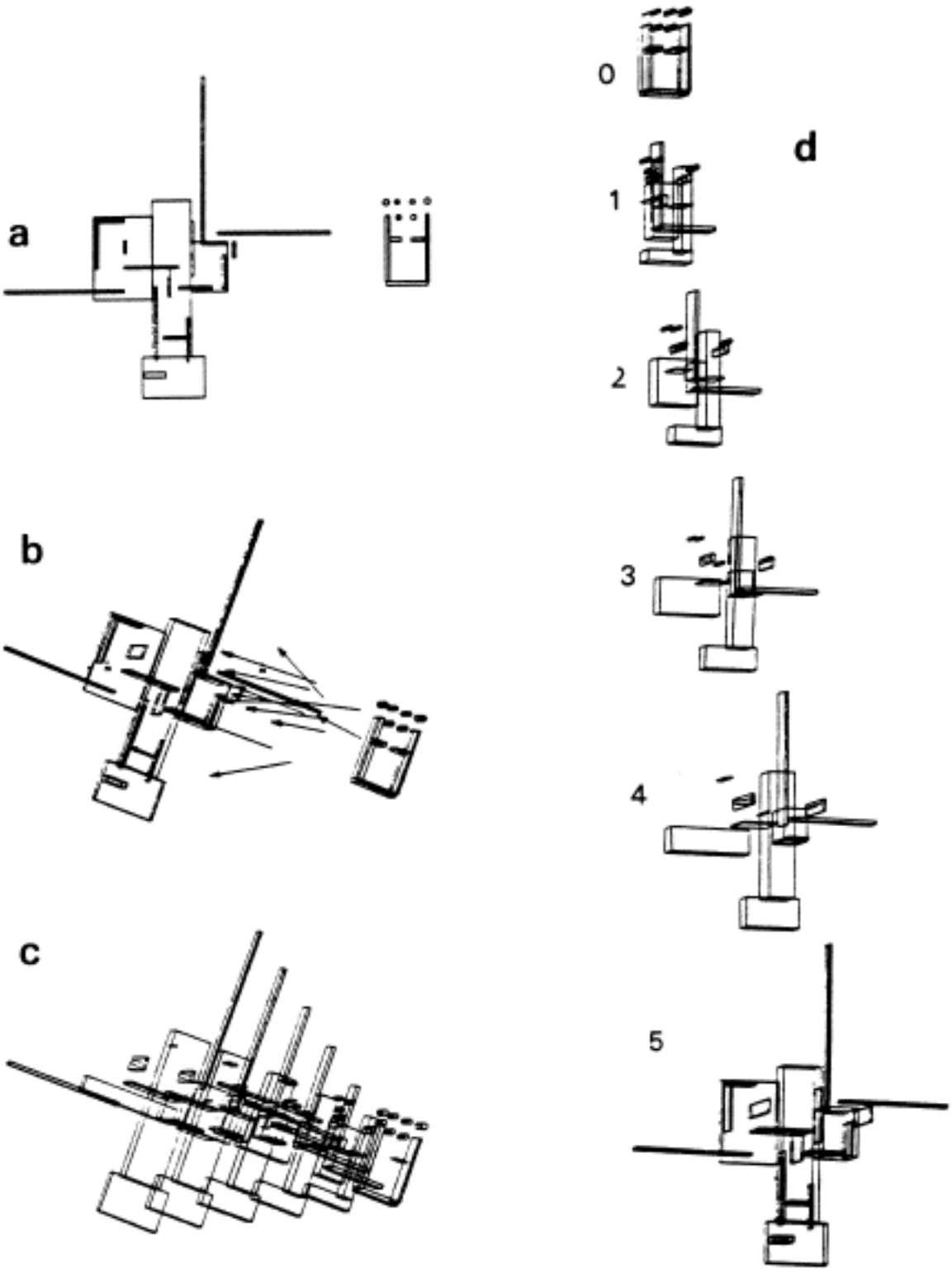


**Figure 10. Use of reformations to cross the forms of (a) a Doric Temple and Mies van der Rohe's Brick House. (b) the pairings and the directions of the reformations. (c) Overlapping in-between states of the reformations. (d) From the Doric Temple to the Brick House in five steps.**

executed in two different fashions. In 10(c) the end forms (initial and destination) are positioned at a relatively close distance, resulting in overlaps of the four in-between states. Retaining all in-between instances as a single composition leads to design schemes quite different from selecting in-between forms, one at a time, as shown in 10(d).

## 7. CONCLUSION: IS THERE AN ANLOGY WITH MUSIC?

This paper presented an experimental design tool implemented in a CAAD system. It currently runs on an IBM RT under a Unix environment. Its graphics support is provided by X Windows. The main feature of the tool is the dynamic transformation of 3-D objects. In brief, the system offers to the user/architect the means to create and edit architectural models, transform them from one position or form into another and finally render them.

This project attempts to introduce the factor of *time* in the architectural design process and to treat architectural objects as tentatively soft and elastic entities, in search for an appropriate manifestation of a form. It takes advantage of the ability of the computer to dynamically display images in real-time, and uses it as a design tool. The user is placed in the position of a director and the whole design becomes a process of synchronization and orchestration of moving and changing forms.

Throughout this discussion, the terms orchestration, composition and synchronization have been frequently used. It seems that, intentionally or not, there may be an analogy and a similarity with music.

Originally, there was no intention to give musical characteristics to the reformation process. However, as the development of the system proceeded, borrowing analogies from music became inevitable. That analogy appears even more appropriate when more than one objects are in motion. They need to be somehow synchronized whenever, for example, their speeds are different. Thus issues which are more common in music than in architecture had to be addressed. The thought also arose whether analogies other than time, could be applied as well; for example, melody, rhythm or harmony. The idea of orchestration became evident, first taken, of course, from music, and was used to describe the action of selecting objects, assigning tasks and evaluating performances. The conductor's baton became the cursor's movement on the screen indicating time beat and spatial viewpoint. Cords were derived through assignment of parallel movements of objects as if they were parallel voices and a pause was simply the transformation of an object to itself

remaining temporarily in it's own position. Without question, these analogies deserve further investigation.

The work presented in this paper is actually a hypothesis that reformations could prove to be valuable architectural design assistants. To completely verify this hypothesis, they will have to be made available by some commercial CAD system, so that they may be tested in real life projects. In conclusion, we can only hope that this paper may persuade some CAD manufacturer to provide them.

## ACKNOWLEDGEMENTS

This project was supported in part by National Science Foundation Project # DMC-8609893. I wish to acknowledge the contributions which my advisor Dr. Chris I. Yessios made to the project and this paper. Quite a few of the original ideas are actually his. I remain responsible for the way they were implemented.

## REFERENCES

Ammer, C. [1972] **Harper's Dictionary of Music**, Harper & Row, New York.

Cole, H. [1974] **Sounds and Signs**, Oxford University Press, London.

Eisenman, P. [1986] "The Futility of Objects", **Harvard Architecture Review** 3, p. 66.

Evans, R. [1986] **Not to be used for wrapping purposes**, AAFiles 10, p. 70

Libeskind, D. [1981] **Between zero and infinity**, Rizzoli.

Tschumi, B [ ] "Madness and the Combinative", **Oppositions**

Yessios, C. [1987], "A Fractal Studio", **ACADIA 87 Proceedings,** North Carolina State University, The Association of Computer Aided Design in Architecture.