Towards an Understanding 3D VR Architectural Design System *

Jarke J. van Wijk
Eindhoven University of Technology
Fact. Mathematics and Computer Science
vanwijk@win.tue.nl

Bauke de Vries
Eindhoven University of Technology
Fact. Building and Architecture
b.d.vries@bwk.tue.nl

Cornelius W.A.M. van Overveld
Eindhoven University of Technology
Fact. Mathematics and Computer Science
wsinkvo@win.tue.nl

Abstract

Current systems for Computer Aided Architectural Design are of limited use for the early phase of a design. We give a specification of a new system, in which we strive for a more natural and intuitive interaction. A key element is geometric analysis. From the input of the user, implicit geometric and structural relations are derived, which are used during subsequent interaction.

Keywords: Architecture, CAAD, VR, Interaction, Constraints

1 Introduction

Last 30 years many computer graphics systems for architectural design have been developed. The complex models and beautiful images that have been produced give the impression that Computer Aided Architectural Design (CAAD) is well-established and succesful. However, in the day-to-day practice of architects its application is limited [3]. Two main reasons can be discerned: First, most systems are far too complex to use by designers, second, the support that these systems offer for the early phases of the design is highly limited. As a result, Computer Aided Architectural Design is mainly used for the final phases of the design, as a sophisticated alternative for the drawing-board.

We think that the usability of a design system in the sketch phase can be improved if as much as possible information is extracted from the input of the user and reused during the design. In this paper we propose a partial solution: geometric analysis. The design is continuously scanned for geometric pecularities, such as coincident faces, orthogonal lines, etcetera, which relations are maintained as much as possible.

In section 2 the background of this work is described: CAAD, VR for architecture, the VR-DIS project, and a global data model for architectural design. Geometric analysis is described in section 3. Starting from an example, we introduce *implicit relations* as a means to improve the interaction. The embedding in an interaction cycle is described in section 4. From the gestural input of the user and implicit relations multiple possible modifications of the geometric model are derived, from which the user selects the most appropriate. Finally, in section 5 future work is described.

2 Background

2.1 CAAD

Most CAAD packages developed from 2D drawing programs. The emphasis for the application of these packages is on the final design stage when accurate models of the design are created and finally floor plans, cross sections, facades, etc. are plotted. Analyzing these systems two main approaches can be recognized for creating 3D models:

1. The drawing is primarily established in 2D (e.g. a floor plan) and then extruded into the third dimension. The original 2D design is lost and cannot be edited anymore as such.

*Proceedings of the 15th Twente Workshop on Language Technology, "Interactions in Virtual Worlds", May 19-21, 1999, Enschede, The Netherlands, p. 219-224.

2. The drawing is primarily created in 2D with drawing entities that already have a 3D representation. The 3D representation is shown separately and can only be edited via the 2D representation. Often to update the 3D presentation of the design the CAD application has to be triggered explicitly.

CAD users and developers try to find ways to use CAD systems not only in the final stage of the design process, but as a design support tool during all design process cycles. More precisely, architectural designers demand design tools that will support their creativity.

Since about five years CAD packages shifted or extended to the early design stage focussing on conceptual models represented by 3D geometries. A common approach in the more recent generation of design tools for manipulation of 3D geometries is by defining building element types with a set of parameters and instantiate them in a design with specific parameter values. In some CAD tools this is a one-direction process, i.e., the geometries cannot be changed afterwards by updating the attribute values. The advantage though is that one can edit the geometry at a lower level (e.g. edges and vertices) creating a desired shape which could not have been achieved by changing the building element attribute values. Other CAD systems (e.g. ArchiCAD) maintain the relationship between parameters and geometry. These parameters are abstracted from a category of building elements defining all variables that can be modified within certain ranges (e.g. length, color, etc). Yet another way to provide a flexible 3D-design tool is by describing a specific shape as a stack of operations (e.g. 3D Studio). After creation of the geometry primitive, every operation with its accompanying parameters is stacked on top of the last one. Every parameter can be changed afterwards as well as the order of the operations. Though the approach is very powerful, it requires a very good understanding of the operation structure. Design strategies for creation of form that are still in the research state are the use of shape grammars and rule-based design. Instead of abstracting from building element geometries, here the geometry creation process is formalized.

2.2 VR for architecture

The application of Virtual Reality until now is mainly restricted to realistic visualization of shapes. VR software (such as World Tool Kit and Device) is optimized for generating pictures of a building model consisting of objects with a shape representation and a texture mapping while navigating through the model. Moreover, objects can exibit some behavior like gravity and collision detection. With these capabilities a very realistic impression can be created of a building which only exists in the mind of an architect. The architect himself/herself or the principal can use VR to judge a design on its esthetical and its functional qualities. The display of the design in VR is dynamic, but the design itself is static. To take VR one step further the designer must be able to interact with the design by creating, modifying and deleting design objects and by evaluating the design to certain aspects. Current day-to-day pratice for generating a VR model of an architectural design consists of the following sequence:

1. Create a 3D model (e.g. by Autocad);
2. Attach material properties and/or textures to surfaces (e.g. by 3D Studio);
3. Add light sources and execute lighting rendering (e.g. by Lightscape);
4. Import the 3D model with appropriate surface rendering in the VR environment (e.g. World Up);

From the sequence only step 3 may be omitted if accurate lighting is not considered necessary. After once having created a VR model, making alterations to the design is a rather tedious process since it requires re-execution of the sequence.

2.3 VR-DIS

At the Eindhoven University of Technology a research program called VR-DIS is established to pursue an innovative multidisciplinary design system. We envision a design environment which supports the designer in his/her creativity and in evaluating the design. The design is presented in a interactive environment using VR technology. Tools for shape manipulation are offered depending on the kind of operations (e.g. creating openings). Design information (e.g. mass, cost) is represented in a complex data structure which is manageable in the VR environment as well. Already in the conceptual design stage architectural, physical, structural, and engineering knowledge (from the past) can be consulted and adapted, to improve building efficiency and quality. For collaborative design, building participants can meet in the 'design space', discuss the design and make alterations. Last but not least the user will be invited to take part and give insight about user behaviour and user satisfaction. In the VR-DIS research program there is an emphasis on conceptual design. There are two main reasons to focus on this design stage:

1. Important design decisions are made in the early design stage based on tentative, incomplete and inexact knowledge. This notion has been recognized by many researchers. Our approach towards this problem is in trying to find general proven design and engineering concepts and make them available to the designer who is faced with a new design problem.

2. VR technology performs best in the early design stage. Comparing traditional Windows technology and VR technology against a range of functional aspects of architectural design systems indicates that Windows technology performs best in the final design stage using a 2D representation, whereas VR technology performs best in the early design stage using a 3D representation [6].

The VR-DIS research program results in a platform for CAAD that is implemented incrementally. In this paper we present and discuss our current ideas and directions for further research with respect to geometric interaction of the architect with the system.

2.4 Model

The VR-CAAD system we develop will allow for the creation and manipulation of geometric entities at different levels of abstraction. We distinguish three layers in the system, namely:

1. *Product model description.* On this level the geometry of a building element is described using a semantically rich syntax. This means that the geometry and its relationships are described in terms of the corresponding building elements, such as walls, floors, etc. Product models consisting of type definitions and their instances are usually implemented as design databases. Typically the available data can be processed by domain specific applications (e.g. Finite Element Analyses).

2. *Geometric objects.* On this level geometry is described with primitives such as cuboids, rectangles, cylinders, etcetera. Parametrized geometries of building elements are built on top of these primitives, they are the basic entities used to create any kind of (architectural) shape.

3. *Geometric elements.* On the lowest level the geometry is described as vertices (points), edges (lines), and faces (polygons). Polygons are easy to process by computer graphics hardware to obtain maximal frame rate performance. User interface interaction like selecting, dragging, etc. is typically executed through identification and recalculation of points, lines, and faces.

The challenge for the development of the VR-CAAD system is that the designer wants to communicate on the product model layer, while on the other hand he/she wants to interact on one of the lower levels. In the following sections we discuss how this can be done.

3 Geometric analysis

On each of the previously defined layers the information can be modelled as a set of objects, where each object has a type, parameters, and relationships with other objects. For instance, a building element can be a wall with a certain thickness that separates two spaces; a cuboid is a geometric element with certain dimensions that belongs to a particular group of elements; a vertex has coordinates and occurs in edges and faces.

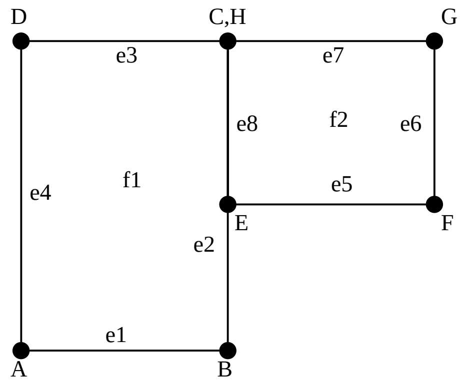

Figure 1: Two rectangles

As a simple 2D example, figure 1 shows a very simple floorplan, consisting of two rectangular spaces. The corresponding data structure for the two lowest levels is shown in figure 2. On each level a complete geometric model can be defined with a minimal set of information, for instance just by enumerating the geometric objects and their position, dimensions, and orientation. During interaction all levels have to be kept consistent with each other, which is often not trivial.

Suppose that the architect modifies his model via manipulation of vertices, lines, and faces. A key question is:

> *Given a certain manipulation, how can we select the most meaningful realisation of this manipulation?*

For example, suppose that the user moves point C in figure 1 in vertical direction. This can be interpreted in various ways:

1. The whole design must be translated;

2. Vertices D, C, G, and H must be translated;

3. Vertices D and C must be translated.

The most probable interpretation here is option two. The user has moved a point, which indicates that he wants a local operation, hence option one is probably not desired. Option two now keeps most of the structure intact. However, we are not certain, and there are situations where the choice is less clear. How can we make the right choice?

We expect that geometric analysis can provide a partial, though significant answer. In a geometric model many subtle relationships can be discerned. If we look at the example, we see that the rectangles touch each other, that vertex C and H coincide, that vertex E is located on edge e2, that e2 and e8 partially coincide, and that e3 and e7 are in line with each other. We call such relationships *implicit relations*. They emerge when a user builds up and modifies his model.

We assume that such relations are not coincidential, but reflect the intention of the architect. For instance, in the product model relationships between building spaces or building elements are typically described in terms derived from natural language such as, next to, on top of, etc.

Many different types of implicit relations can be defined, in which two or more elements of the same or different types participate. As an example, some relations between two rectangles are shown in 3.

We can now reformulate our original question as:

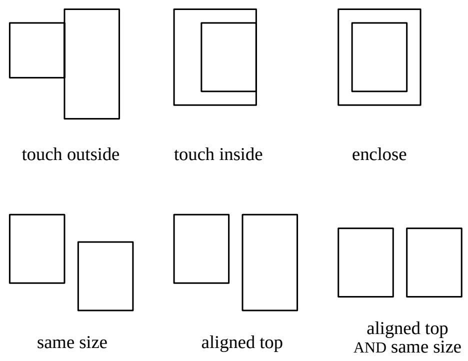

Figure 3: Relations between rectangles

> *Given a certain manipulation, which implicit relations must be maintained, disabled, and added?*

Implicit relations that are important for the user must be recognized and used as invariants, possible at the expense of less important implicit relations that are disabled. Furthermore, new patterns in the mutual relations between objects must be recognized and added to the implicit relations. Note that implicit relations are redundant. Both the representation in terms of geometric objects, as well as the representation in terms of vertices, lines, and faces are already complete in themselves.

In the ideal case this whole process is automated. The system continuously extracts as much information from the input of the architect as possible, and reuses this information during subsequent interactions. Or, formulated differently, we aim at a system that understands what you mean. Having said that, we realize that this goal is hard, if not impossible, to reach.

The major problem is the detection and weighing of the implicit relations. Which ones are meaningful, which are not? Given multiple solutions, how to find an optimal one? These questions often cannot be answered without active participation of the user. Intelligent systems have often a habit of making the wrong guess. In the next section we discuss how interaction can benefit from implicit relations without being bothered by misinterpretation on the side of the system.

Another problem is to find solutions numerically that satisfy a certain set of implicit relations. We expect that local propagation is a viable route for this. Starting from the object that is manipulated, its neighbours are considered, if not all relations can be handled satisfactorily, their

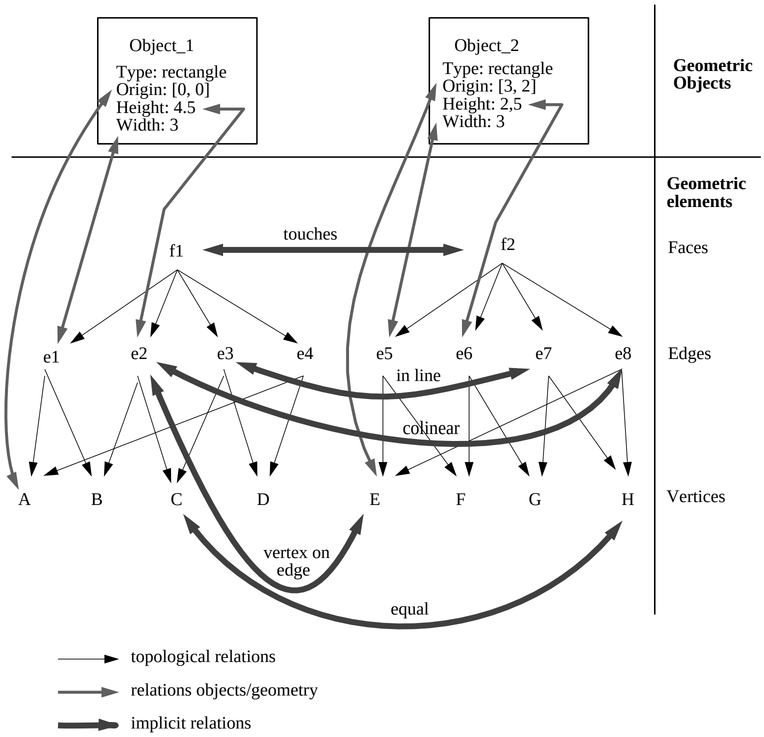

Figure 2: Data structure

neighbours in turn, and so on.

4 Interaction design

The system that we intend to build should be as platform independent as possible. To explore the limits a large set-up (for instance a CAVE [2]) would be ideal, however, for application in practice by architects, it is more realistic to assume not much more than a PC, possibly extended with stereo graphics. Projection based systems, such as a Responsive Workbench [4] could be an attractive compromise. Anyhow, the method we propose here is independent of a particular hardware environment.

In any environment, the design of the interaction is critical. We aim at a system where the user inputs and modifies his design mainly via gestures, to approximate the natural way an architect designs a building via sketching as close as possible. Overviews of inspiring work in this area can be found in [1] and [7]. These gestures are interpreted by the system. The type of action and the object acted upon are derived from the shape and scope of the gesture. Also here understanding plays a major role: The system has to determine the most probable interpretation.

The main actions are creation, transformation (translation, rotation, and scaling) and deletion. New objects are created by sketching on surfaces. If the scope is taken into account a few strokes suffice to define the type and the dimensions of new objects. A , -shape drawn on a wall defines either a window or a door, dependent on the position of the lowest point; a straight stroke on a floor defines a wall, a small circle on a floor a column.

Transformation is probably the most important action. A natural way to support design is by adapting parts of other existing designs into a new situation by means of (geometric) transformations. Also, during the design the architect wants to consider many variations of dimensions and compositions of the building objects. A well-known way to implement this is to let the user select an object upon which a control-box with handles appears. The user can select a control-point and drag it. Another method is to let the user manipulate individual vertices, edges, and surfaces directly. In principle this is more efficient, because the user can perform such manipulations with a single gesture, without the separate selection step of the standard method, but this has to be confirmed by experiments.

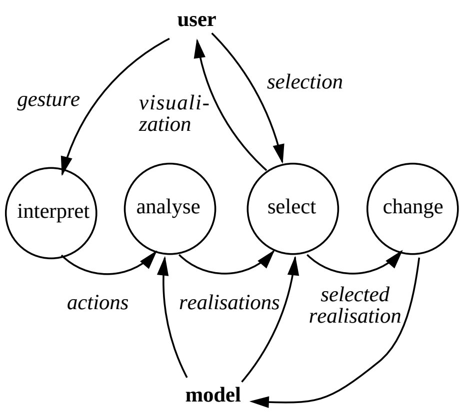

Figure 4: Interaction cycle

Deletion can be specified rather easily: The user erases a part of the design by making a Z-stroke on its surface.

The result of a gesture is one or more low-level actions, such as "Create a door in this wall at roughly this position" or "Move this vertex by a certain distance in a certain direction". Multiple alternative actions can result when the gesture is ambiguous. In the next processing steps these actions are translated into changes of the model. Figure 4 shows in a data-flow diagram how we envision this. The actions are analyzed based on the implicit relations, which leads to several different realizations.

We foresee two ways to proceed. The first is to extract more information from the gestures of the user. A slow gesture could indicate a global change, while a fast movement could mean a local change . With a fast movement the user breaks up a connection. Also, two-handed input can be used. The user fixes with one hand a part of the design, while translating another part with his other hand.

The second, and most important way to resolve ambiguities is to let the user decide. We do not believe in infallible intelligent systems. In the end, the user is the only one who can decide what's the right interpretation. To support him, multiple alternatives are shown, from which the user picks the right one. This can be implemented in various ways. Alternatives can be shown sequentially, upon which the user confirms this alternative or steps to the next. Alternatives could be shown simultaneously in separate windows. Such a solution could work in a desk-top setting, but requires much screen space, and requires the user to shift his focus. A challenging solution is to show all alternatives simultaneously in the same view, for instance as line drawings with different colors. The user can then pick the right one by specifying the color. Speech input seems a natural choice here.

After the selection of a particular realisation, the system updates the model: the geometry is changed and implicit relations are added or disabled.

5 Discussion

We have presented an overview of our ideas for a Computer Aided Architecture Design system. The central theme is to provide the designer with a natural and understanding system, such that he can focus on his design task. To realise this, we propose to extract and reuse as much information as possible from the input of the designer:

- The shape and scope of the gestural input of the user is analyzed;

- Via geometric analysis a set of implicit relations between geometric elements is maintained and applied during the interaction.

Our next step will be the implementation of the system. This will be a large task, during which many details have to be elaborated, and new problems and directions for extensions will show up. Also, we realize that it is hard to predict which choices are most natural and understandable in practice. We therefor will develop the system cyclically and iteratively. We start with a limited shape domain, possibly even 2D, to evaluate the basic algorithms and test their usability. In next phases we will gradually expand the shape domain and other functionality.

During this next phase we will try to find answers to several open questions. For instance, we are not sure if it will be possible to hide the implicit relations completely from the user. If not, the next question is how to visualize and let the user interact with them. Another large area for future research concerns the shape domain, especially the integration of free-form surfaces within this concept.

References

[1] Billinghurst, M. Put that where? Voice and Gesture at the Graphics Interface. *Computer Graphics* **32**, November 1998, p.60-63.

[2] Cruz-Neira,C., D.J. Sandin, T.A. DeFanti, R.V. Kenyon, and J.C. Hart. The CAVE: Audio Visual Experience Automatic Virtual Environment. *Communications of the ACM 35,* 6, juni 1992, pp. 65-72.

[3] Dorsey, J., and L. McMillan. Computer Graphics and Architecture: State of the Art and Outlook for the Future. *Computer Graphics* **32**, February 1998, p. 45-48.

[4] Krüger, W. et al. The Responsive Workbench: A Virtual Work Environment. *IEEE Computer 28,* 7, juli 1995, 42-48.

[5] http://www.ds.arch.tue.nl/Research/program

[6] Vries, B. de, and H.H. Achten. What offers Virtual Reality to the Designer? In: *Proceedings of the Int. Conference on Integrated Design and Process Technology*, Berlin, Germany, 1998.

[7] R. Zeleznik. Sketching in 3D. *Computer Graphics* **32**, November 1998.