Space Pen
Annotation and sketching on 3D models on the Internet

Thomas Jung, Mark D. Gross, Ellen Yi-Luen Do
Design Machine Group, Department of Architecture, University of Washington, Seattle, WA 98195-5720, USA

Key words: Collaboration, Annotation, Java3D, VRML, and Sketch.

Abstract: Designing a building or a new urban space is collaborative work that involves several people with different backgrounds. To achieve consensus all participants in the process meet to discuss documents such as floor plans and sections. This paper reports on the progress of Space Pen, a new system to allow several users to draw on and annotate a three-dimensional representation of a building remotely over the Internet.

1. INTRODUCTION AND MOTIVATION

1.1 A 3D collaborative environment

Communication between firms and their clients and contractors is becoming increasingly challenging as many projects now involve people and teams from different countries or even continents. The only way for everyone to participate in the design throughout the whole process is to schedule meetings, which can be difficult to set up and therefore months apart. For a client it can be frustrating not to be constantly part of the progression of the design. Yet it is often time consuming for an architect to reconsider and amend decisions that the client might not agree on, or that are too difficult or expensive for the contractors to realize.

One key to improving online communication is to make the project as clear as possible to all the participants. Floor plans and sections are useful to communicate ideas among building professionals, but people without architectural training have difficulty envisioning space and volumes from 2D documents. Where before scale models were often used to communicate design ideas with clients, now most architectural firms have the capacity to build 3D models and present computer rendered images to their clients.

When Disney created California Adventure Park, designers created a full 3D virtual model of the design. Disney's officials examined the model, which revealed the building's structure and the different phases of construction in a virtual environment. According to Disney, problems and design issues, totally unobvious from the physical model and 2D documents, were discovered after seconds of browsing around the virtual model (New York Times, 2001).

In this paper we present Space Pen, a pen-based reviewing system that takes advantage of the Web. Space Pen enables clients, contractors or associates to view a 3D representation of the building (or urban space) and annotate those models by writing and drawing on them anytime, anywhere, on their ordinary Internet browser throughout the entire design process.

1.2 A pen-based interface to sketch in 3D

Online critique and collaboration involves people with different backgrounds and different levels of confidence in using computers. A typical architectural firm usually deals with several distinct clients and contractors each year, all with various computer skills and knowledge. Drawing and sketching communicate ideas quickly, clearly and easily. A pen-based interface to a CAD program eliminates the need for great knowledge of structured menus or command line interface, and for precision in the execution. As Garner mentioned in DCNet'00 (Garner, 2000), "the widespread use of sketching in design activity would seem to suggest that it offered appropriate support to both the defining and resolving of design problems".

Space Pen uses a (digital) pen as a single input modality for annotations and drawings. Because pen and paper is the most traditional and intuitive way to draw and review documents, we can expect every user of our system to understand immediately how to leave comments and annotations.

In Space Pen sketches made on the pen tablet are translated into 3D by leaving digital ink onto the nearest surface of the model. The marks remain on the surface and can be seen from any viewpoint as a visitor walks virtually around the annotated surface.

1.3 Related Work

Online collaboration and annotation on 3D documents is a wide area of research that has application in various fields, especially in architecture. We briefly describe below related work in the use of 3D VRML (Virtual Reality Modeling Language) for design collaborations on-line annotation systems, and 3D input interfaces developed by both research laboratories and commercial software firms.

Christopher Peri at UC Berkeley taught students collaborative design by interacting in a VRML world instead of chipboard models (Peri, 2000). His experiment demonstrates that VRML models can replace physical models to support a discussion of design issues in a project.

Web PHIDIAS (McCall, Holmes, et al., 1998) enabled users to annotate 3D VRML models on the Web using the PHIDIAS hypermedia system to organize the comments. Craig and Zimring's system (Craig, Zimring, 1999) annotated a 3D virtual courthouse with symbols such as arrows or circles. Like Space Pen these systems use 3D VRML models to support collaboration over the Internet. However, none of these systems can import VRML models directly from standard 3D-modelers, and in order to use them, an architecture firm would require extra personnel to perform VRML translations and encoding.

The ProjectPoint (ProjectPoint) and the WebOrganic (WebOrganic) web sites provide services to leave comments and annotations on 2D documents and on Web pages using a regular Internet browser. WebOrganic is more generic; it can be used to comment on any HTML page. ProjectPoint is more architecturally oriented and lets people comment and annotate 2D documents such as floor plans and sections using a browser. The ProjectPoint web site assists the architectural firm throughout the design and construction process by providing allocated server space to share, comment or annotate documents. ProjectPoint is 2D, but it can use Autodesk's VoloView Express to share 3D models.

Autodesk's VoloView (VoloView) and Sigma's eZ (eZ) let you annotate 2D or 3D objects online. However they have little or no walkthrough capabilities and annotations in 3D are actually made in 2D (on a transparent plane in front of the viewer). Both applications set up their own complex environment to provide sophisticated annotation support. They are powerful if the users spend time to learn how to manipulate the models and add annotations. However, we believe drawing-based input will be more intuitive and easier to use for people who don't have time or desire to learn another complex computer application.

Various research projects have investigated sketching as an interface to modify and create 3D models. SKETCH (Zeleznik, Herndon, Hughes, 1996) is a 3D modeler that uses strokes made with a 3-button mouse to create geometry in the 3D space. SKETCH only understands a finite number of strokes, which makes its drawing environment less intuitive than sketching with a pen on paper. Teddy (Igarashi. Matsuoka, Tanaka, 1999) creates freeform models and rounded objects with a simple and straightforward pen-based interface. For example, you draw an oval to create 3D bubble that users can view and rotate, and/or sketch on top to add more objects, but it cannot create the rectilinear objects that are more often needed in architectural practice. STILTON (Turner, Chapman, Penn, 1999) on the other hand is mainly designed to sketch and create 3D objects made of straight lines over an existing VRML model. With STILTON, unlike Space Pen, the user doesn't draw directly into the 3D space, but on a transparent surface in front of the viewer. The 2D drawing can then be converted into a 3D object and is automatically placed in the 3D world.

These are powerful systems to create 3D geometry from sketches, similar to what designers would do on paper. However none of them combine an easy-to-use pen-based interface in a shared collaborative environment. This is the goal of Space Pen. Space Pen also can import any kind of VRML model without further manipulation, and because it's been programmed in Java3D it will work on any Java3D enabled browser without additional software or hardware.

1.4 Our previous work

Space Pen extends a project we have been working on in the past two years. That project, called Immersive Redliner (Jung, Do, Gross, 1999), is an annotation system to support collaborative review of 3D models on the web. Each user has the possibility to leave a PostIt™ style note (indicated in 3D by a colored sphere) directly on the model. Each sphere is associated with a comment left by the user and both are saved on a remote server. We tested that system on a real architectural project (Jung, Do, 2000) and learned that leaving text comments was not always sufficient to clearly express a reviewer's idea. Often, a sketch or a drawing on the object in question would have helped to clarify and augment the text annotations.

Redliner was implemented in VRML and EAI (External Authoring Interface) and this implementation architecture sharply limited further development. We decided to port the whole system to a much more powerful, new software platform (Java 3D) that would especially support three-dimensional drawing as input.

1.5 Scenario

Space Pen works directly on any regular web browser with Java 1.3 capabilities. When it is completed, architects, clients and/or contractors will be able to use it to review and comment on a 3D representation of the architect's design proposal. Participants log on to the Space Pen web site to view the project in 3D in their own web browsers. They move around the building as they would once it has been built and leave drawing marks and annotations directly onto the 3D model, and save those as part of the model[1]. Each user chooses a different pen color to represent their comments or drawing. Figure 1 shows the Space Pen applet window. Once logged on, the user becomes immersed in the virtual 3D model. On the left is a set of different pen colors for drawing annotations on the model. The user can navigate around the model using the arrow keys of the keyboard.

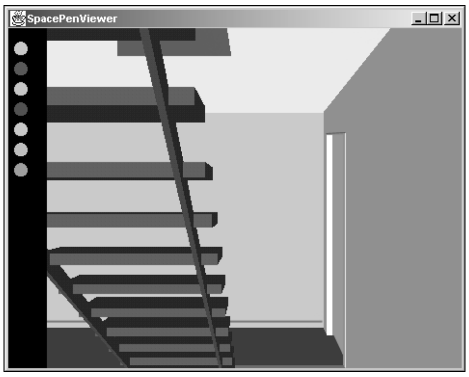

Figure 1. The Space Pen window. The user logs in and is transported inside the model.

Figure 2 shows the annotations drawn by the first user who suggested separating the window in the model into two separate panes. Because this proposed change doesn't need much explanation, the text can be written graffiti-style, directly on the problem area.

[1] At the time we are writing this paper, the saving part of the prototype has not been implemented yet, but using the same technology we used for Redliner, we believe that it can be completed easily.

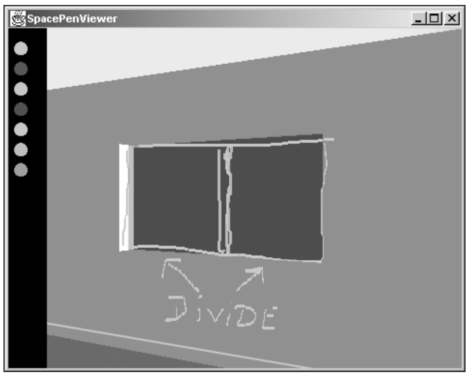

Figure 2. Drawing marks left by the first user.

A second user logs in (Figure 3) and suggests creating two windows instead of one by moving the right windowpane towards the wall. The drawing marks left by each user are embedded directly in the 3D model with different colors and can be seen from different points of view.

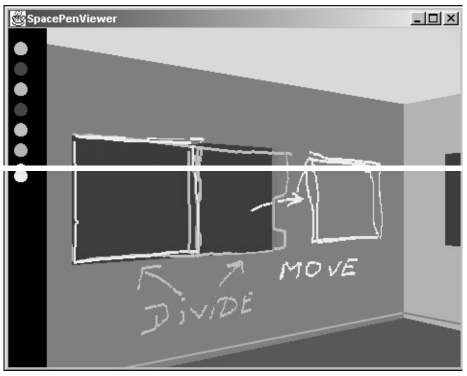

Figure 3. Drawing marks left by a second user on top on the first ones.

Figure 4 shows the comments made by a third user who decided to leave drawing marks as well as a typed-in note associated with a more elaborate text. The line-width of the marks in the 3D model is independent of how far you're standing from them (Figure 4, right). Even from a distant point of view, the marks appear clearly on the building.

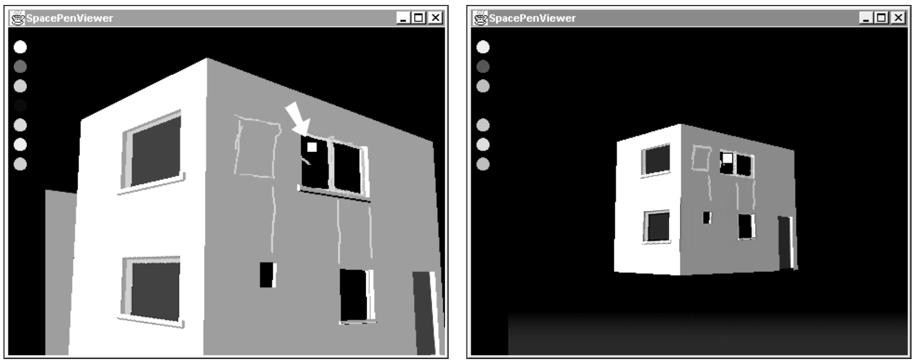

Figure 4. More elaborate comments can be left and will be represented as a PostIt™ note stuck on the model (left, pointed by the arrow), and can be seen even from further away (right).

This simple scenario demonstrates how easy and straightforward the Space Pen interface is to use and how in a few minutes design suggestions can be recorded graphically and in three dimensions. By walking virtually around the building, every user gains an immediate understanding of the project and of the changes proposed by others. Using Space Pen, design issues can be discussed and proposals can be made directly into the 3D environment even if all parties are miles apart from each other.

2. IMPLEMENTATION

Space Pen is a Java 3D applet. It is implemented using the Java 3D API, (Application Program Interface) a set of Java classes that has powerful capabilities for displaying and manipulating 3D objects. In the following we describe some basic features and concepts we used to implement Space Pen.

2.1 Basic Java 3D concepts

Although powerful, with a wide range of possible manipulation and interaction with 3D objects, Java 3D has a complicated way to represent fairly simple concepts. The Java3D API has no pre-defined default values for their 3D objects. Every object must be specified explicitly as clickable, collidable, normalized and so on.

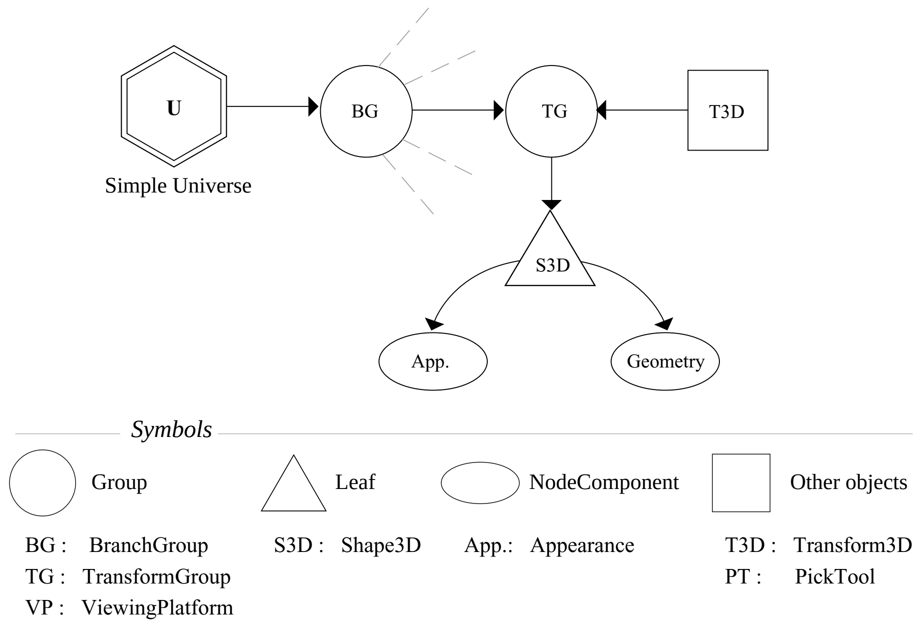

Figure 5. A diagram of a standard Java 3D scene explaining the relationship between each object. On the bottom is a list of symbols and abbreviations used in our diagrams.

Figure 5 shows a diagram of a basic scene implemented in Java 3D. In most Java3D worlds, the top-level object is a Simple Universe (a subclass of VirtualUniverse that sets up the basic environment for Java 3D scenes). The 3D objects are then described in a scene graph composed by BranchGroup (BG) and/or TransformGroup (TG) nodes. Each object has a specific shape (S3D) composed by a geometry and an appearance. More functionality can be added at all levels of the scene graph by adding several other objects. For example a Transform3D (T3D) object can apply transformation matrices on a TransformGroup node. We also used objects like Behaviors or PickTools in the Space Pen implementation.

2.2 The Space Pen implementation

The implementation of Space Pen can be divided into 3 parts:

- Importing and interpreting 3D VRML models
- Drawing and interpreting mouse events
- Walkthrough capabilities

2.2.1 Importing and interpreting VRML models

Although it is widely used, VRML has not become the established standard for 3D on the Internet. However, most 3D modelers can export to this format. Therefore, Space Pen imports and uses VRML models instead of "pure" Java3D models. A VRML Loader Package, part of the Java3D API, imports a VRML model into the Java3D universe.

Nevertheless, we found that a simple import of VRML geometry was not sufficient. We needed to be able to click on an object and have the system to recognize what kind of geometry was selected, the 3D coordinates of the point clicked and eventually the normal of each designated surface. This is not done automatically in Java 3D.

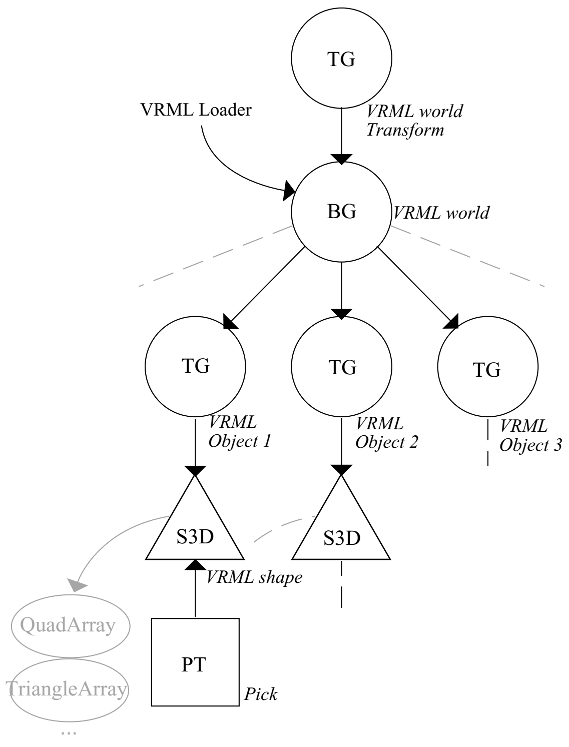

Figure 6. A system diagram showing the importing of VRML objects, the interpretation of each geometry shape and the assignment of Pick interaction.

Figure 6 describes the process of loading the VRML scene, recognizing its shapes and making all the objects clickable (or pickable). In order to be picked (selected), each VRML Shape must be associated with a PickTool object. We need first to count the objects in the VRML world. Then we can loop to analyze each object's geometry. With each geometry (spheres, boxes, cylinders, planes, triangles…) we associate a PickTool object. Currently, the system only recognizes IndexedFaceSet VRML geometry, which most 3D software uses to describe the entire VRML scene.

Immediately after loading the VRML model, Space Pen assigns every surface in the model full intersection capabilities through the PickTool object, allowing users to select an object and enabling the system to detect which object(s) the user is drawing on.

2.2.2 Drawing on 3D models

Space Pen enables users to draw directly on the surface of the model using a mouse or better, a pen. The system associates two different behaviors with two different pen (or in the Java3D terminology, "mouse") events. Dragging, or drawing on the surface of the tablet, draws lines onto the model; a click event initiates a text annotation via the keyboard.

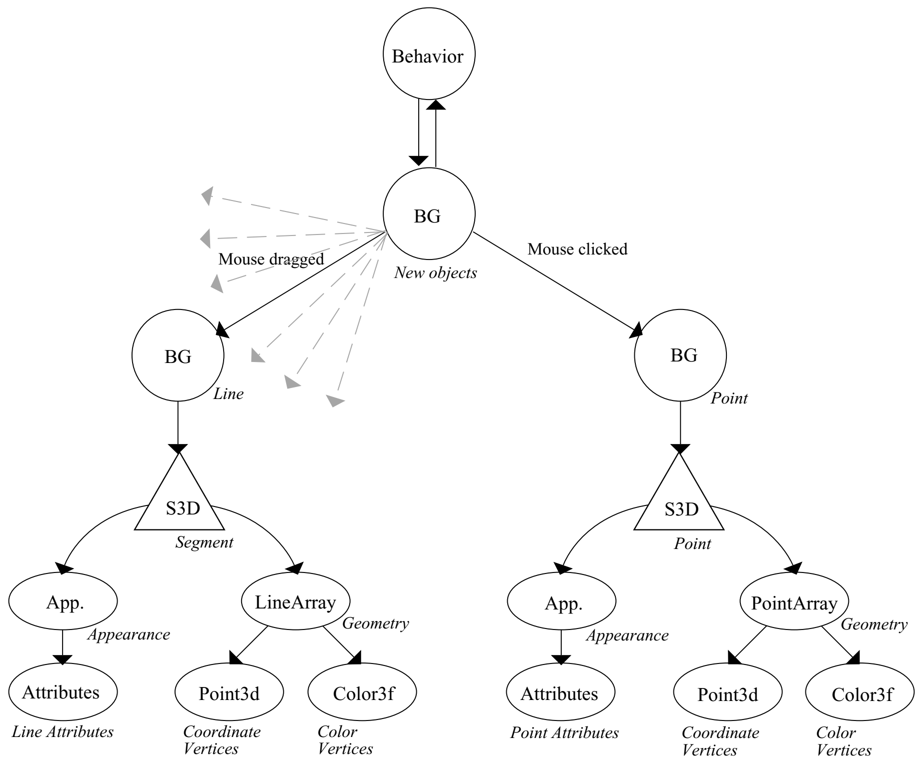

Figure 7. Implementation of the drawing behavior.

The drag method finds the closest intersecting points along the path on a surface. The 3D coordinates of these points are saved in an array and interpreted as key points of a drawing line. The line drawn is a Java 3D object composed of a series of segments defined by 3D coordinate points, color, and boldness attributes.

The click event triggers the leaveComment method that attaches a Point3d object at the click location and brings up a text-input interface for the user to type a more elaborate comment with a keyboard.

Java 3D lines and points have a boldness attribute that sets their width in pixels, independently from how far the user is standing from these objects. This makes comments and drawn annotations visible from most locations in the 3D world.

Figure 7 shows the relationship between all the new objects added into the scene. The behavior object creates a new BranchGroup object that will contain the new lines and comment points. A Segment Shape3D object is created and attached to the Line BranchGroup for each pair of points along the drag path.

2.2.3 Walking inside the model

Java 3D standard methods have pre-set viewing options. The KeyNavigator class and the MouseRotate, MouseTranslate and MouseZoom classes let the user interact with the object in the 3D virtual world. The Mouse behavior classes are used to inspect the 3D model. You can rotate it, move it around or zoom on it using the 3 buttons of the mouse. The KeyNavigator class is a very basic walkthrough Behavior set more adapted for flight simulation than for building investigation (for example, the up and down keys are inverted). None of these were exactly suited to our purpose.

For Space Pen we needed the user to be able to walk through a building or a space, look around and maybe fly. These capabilities were only poorly provided by the existing classes. Therefore we wrote a new VRML browser in Java 3D. The first version of that browser simulates walking and flying behavior controlled by the arrow keys of the keyboard.

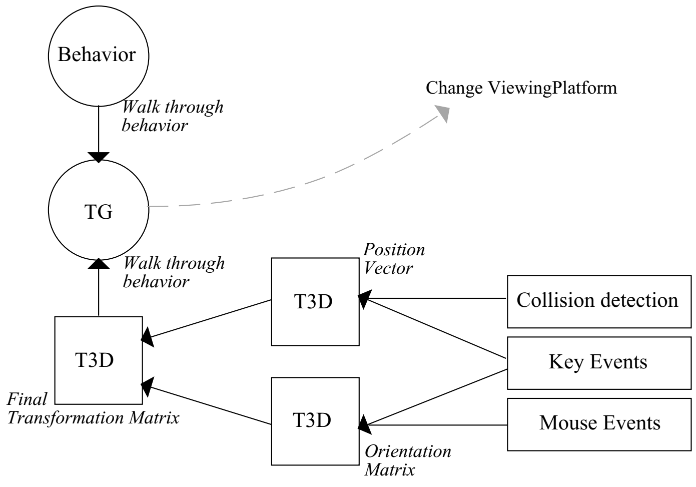

Figure 8. Diagram of the implementation of the walk behaviour class.

Figure 8 explains how we implemented the walk behaviour. A TransformGroup object is created. It takes the result of two transformation matrices as the input, and replaces the ViewingPlatform (an object describing attributes of the main scene graph's view) transformation in the main class. (This class is a derived version of the FlightBehavior class written by Paul Byrne at Sun Microsystems).

Unlike existing VRML browsers, collision detection and gravity are not activated by default in Java 3D. Without collision or gravity, users would start flying when trying to go down and passing through stairs when trying to go up. We resolved that problem by detecting the height of the object immediately in front of the user. If the height of that object is greater than a certain value, then the object is treated as an obstacle and the translation of the user's viewpoint is halted. Otherwise, the object is treated as a step and the viewpoint is translated in the Y direction (up) by a value corresponding to the height of the object.

3. FUTURE WORK

Space Pen is still at an early stage of development. It can load any kind of 3D VRML model in a web browser and visitors can draw or leave notes on these models. We're working on saving annotated models and the integration of all the components: Building on Redliner, we're ready to embed comments, login information and 3D world into a single applet.

However, our goal is more than just reproducing our work with Redliner with sketch input augmentation. We envision Space Pen as a way to manipulate and transform 3D models on the web, using a pen based interface.

Informal observations of people using our system caused us to consider how to make the drawing environment more intelligent. When drawing on non-coplanar surfaces, what we draw is not always what we intended to represent. Our next step will be to process the annotation mark, so the system would recognize which surface the user wants to draw on and what kind of object has been drawn.

We're planning to adapt the 2D drawing recognition technology used in one of our previous projects (Electronic Cocktail Napkin, Gross, Do, 1996) into our 3D Space Pen environment. With symbol and configuration recognition abilities, we could recognize the drawing marks on the 3D as an object or a gesture command for editing and creating 3D geometry. For example, a rectangle drawn on a wall might generate an opening. A rectangle drawn on a horizontal surface could generate a hole or the base of a new column, depending on its size. Symbols such as arrows or textual annotations such as "yes" or "no" could also be recognized and associated with an action to confirm the editing operations. For example, an arrow in an opening might indicate a required dimension; on a door it might assert a move operation with indicated direction and distance.

4. ACKNOWLEDGEMENTS

This research was supported in part by the National Science Foundation under Grant No. IIS-00-96138. The views contained in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

5. REFERENCES

Bimber O., 2000, "A multi-layered architecture for sketch-based interaction within virtual environment", in: *Computer & Graphics, ed. Pergamon*, Volume 24, Number 6, December 2000, p. 851-867.

Brown K., Petersen D., 1999, "Ready-to-Run Java 3D", *Wiley Computer Publishing*.

Craig, D. L. and Zimring, C., 1999, "Practical Support for Collaborative Design Involving Divided Interests", *Media and Design Process, Proceedings of ACADIA '99*, Salt Lake City, p.126-137

Davidson, J. N. and Campbell, D. A., 1996, "Collaborative Design in Virtual Space - GreenSpace II: A Shared Environment for Architectural Design Review", *Design*

Computation: Collaboration, Reasoning, Pedagogy, Proceedings of ACADIA, Tucson, Arizona, USA, p. 165-179

Dorta, T. and LaLande, P., 1998, "The Impact of Virtual Reality on the Design Process", *Digital Design Studios: Do Computers Make a Difference? ACADIA Conference Proceedings, Québec City, Canada*, p. 138-163.

EZ, http://www.ezmeeting.com/

Garner S., 2000, "Is Sketching Relevant in Virtual design Studios?", *DCNet'00 Proceedings, International Journal of Design Computing,* http://www.arch.usyd.EDU.AU/kcdc/journal/vol3/dcnet/garner/

Gross M. D. , 1996, "The Electronic Cocktail Napkin - working with diagrams." *Design Studies 17(1)*, p. 53-70.Gross, M. D. and E. Y.-L. Do, 1996. Ambiguous Intentions. Proceedings, ACM Symposium on User Interface Software and Technology (UIST '96). Seattle, WA: 183-192.

Gross, M. D. and E. Y.-L. Do, 1996. Demonstrating the Electronic Cocktail Napkin: a paper-like interface for early design. CHI 96, Conference on Human Factors in Computing Systems. Vancouver, British Columbia, Canada, ACM. Conference Companion: 5-6.

Igarashi T., Matsuoka S., Tanaka H., 1999, "Teddy: a sketching interface for 3D freeform design", *Computer Graphics, Proceedings, Annual Conference Series, ACM SIGGRAPH'99*, p.409-416.

Java 3D, http://java.sun.com/products/java-media/3D/index.html

Jung, T., Do, E.Y. and Gross, M.D., 1999, "Immersive Redlining and Annotation of 3D Design Models on the Web", *Proceedings of the Eighth International Conference on Computer Aided Architectural Design Futures, Atlanta, USA*, p. 81-98.

Jung, T. and Do, E.Y, 2000, "Immersive Redliner: Collaborative Design in Cyberspace", *ACADIA'00 Proceedings, Washington D.C., USA*.

McCall, R., 1998, "World Wide Presentation and Critique of Design Proposals with the Web-PHIDIAS System", *Digital Design Studios: Do Computers Make a Difference? ACADIA Conference Proceedings, Québec City, Canada*, p. 254-265

Peri, C., 2000, "ARCHVILLE: A Pedagogy for Teaching Collaboration in a VR Environment", *Proceedings of the Collaborative Virtual Environments, San Francisco, USA*, p. 211-212.

ProjectPoint, http://www.buzzsaw.com/content/services/demo.asp

Sun Microsystems Java 3D Engineering Team, 2000, "Java 3D API Tutorial", *Sun Microsystems*, http://developer.java.sun.com/developer/onlineTraining/java3d.

Taub E., 2001, "Small Worlds to Create Bold, New Ones", *New York Times, March 1, 2001*, p. D7.

Turner A., Chapman D., Penn A., 1999, "Sketching a virtual environment: modelling using line-drawing interpretation", *Proceedings of ACM Symposium on Virtual Reality Software and Technology*, p.155-161.

VoloView, http://www3.autodesk.com/adsk/index/0,,224835-123112,00.html

Weborganic, http://www.weborganic.com/

Zeleznik R. C., Herdon K.P., Hughes J.F., 1996, "SKETCH: An Interface for Sketching 3D Scenes", *Computer Graphics, Proceedings of SIGGRAPH'96, Annual Conference Series*, p.167-170.