VR Sketchpad

Create Instant 3D Worlds by Sketching on a Transparent Window

Ellen Yi-Luen Do
Design Machine Group, Department of Architecture, University of Washington, Seattle, WA 98195-5720, USA

Key words: pen-based interface, freehand sketches, diagramming, transparent window, Virtual Reality Modelling Language (VRML)

Abstract: This paper describes VR Sketchpad, a pen-based computing environment for inputting and locating 3D objects in a virtual world. Designer can use the transparency layers to quickly trace and extract any image underlay from other application software. The 3D scene generation has three levels of complexity: simple extrusion of any drawn lines of shapes (i.e., straight or curved wall and column extrusion), solid modelling from a given geometric object representation (spheres, cones and boxes), and complex configuration with objects from graphics library (furniture layout).

1. INTRODUCTION

As on-line communication and entertainment enterprises increasingly use three-dimensional geometry, the ability to quickly create 3D content becomes important. Building three-dimensional worlds for the World Wide Web usually involves complicated CAD and modelling software that uses WIMP (windows, icons, menus, and pointers) interfaces. VR Sketchpad gives creators of three-dimensional content a quick and easy way to author 3D virtual worlds by sketching without coding VRML (Virtual Reality Modelling Language) or learning to model using CAD software.

With experience and training, designer can use CAD systems to create accurate models of 3D worlds. CAD systems are good at producing precise 3D models and supporting detailed editing and revision. However, pen-based systems enable designers to communicate ideas rapidly through approximate sketches with low overhead (direct manipulation by freehand drawing).

Sketching requires no precision or specialised knowledge, and makes it easy for designers to draw, evaluate, and explore as well as to revise and correct. Sketching is useful in settings that require a fast turn-around creation-feedback cycle, for example, during conceptual design. These settings include building and product design, as well as designing computer games and virtual worlds for VR applications, on-line exhibits, and cyber communities on the Web.

Providers of 3D content often start conceptual design by sketching with pencil or markers on paper or by building a physical model to simulate the proposed design. Later they use CAD software or game level editors to create these imagined 3D worlds. The process of making a 3D model with structured commands and operations is far more elaborate than using a pen and paper to sketch. A freehand drawing user interface for 3D modelling—as VR Sketchpad illustrates—provides designers more freedom to explore than a WIMP interface.

This paper presents VR Sketchpad, a freehand drawing interface for rapidly generating three-dimensional geometry in VRML format by tracing diagrams on a transparent window over an image source. VR Sketchpad employs a novel approach for creating 3D worlds. It enables designers to simply make diagrams for spatial partitions such as walls, columns and furniture on a 2D floor plan to quickly generate 3D worlds on the Web. For example, a quick sketch of a floor plan of an exhibition hall generates a virtual scene with walls and columns. Immediately, the designer can experience a virtual walkthrough from a Web browser. Preferred viewer positions marked on the floor plan are translated and embedded into the VRML viewpoint control panel, providing a guided tour path. As the visitor browses the virtual exhibit hall, VR Sketchpad displays the visitor's position on the floor plan sketch. This instant feedback from 3D VR world to 2D floor plan supports navigation and understanding of the created space for both visitors and designers.

2. RELATED WORK

Many projects investigate the use of transparent windows as an interface to facilitate information display or object generation. For example, PAD (Perlin and Fox 1993) uses transparent windows to support "zooming" from abstract information to more details. A reader can navigate through the document space by applying transparent pads on top of the area of interest. Toolglass and Magic Lenses (Bier, Stone et al. 1993) adopt a similar approach of having transparent objects carry with them functionality for user interaction. They employ transparency to allow users to construct new

artefacts with overlapping elements. Translucent Patches (Kramer 1994) use freeform shapes for copying and grouping information. The different patches can overlap and interact with each other (e.g., performing mathematical calculation of figures from overlapping patches).

Electronic Cocktail Napkin's trace layer allows selecting and copying of diagram sketches between different layers, it also supports re-arrangement of the layers through a post-up action with thumbnail representation of the drawings (Gross 1994; Gross 1996; Gross and Do 1996). The Napkin program also supports different pen types. Users of the system can leave marks in different thickness and darkness with the pressure and velocity data derived from the pen stylus. Trinder argues that architects use transparent media such as tracing paper and drafting film not only to bring images together, but also to compose configuration with hierarchy (Trinder 1999). His 'transparent medium' investigates using simple tools such as "push and pull" to translate pixel maps between two layers and a "sketching tool" that translate stroke information as different thickness. His empirical studies on twelve test subjects reveal that using transparent layer and sketching are a good way for design professionals to interact with computer software.

Several researchers also conducted usability studies of transparent user interfaces (Harrison 1996; Cox, Chugh et al. 1998). Their findings are encouraging. They found people could use transparent layers easily to shift their focus rapidly between the different views (e.g., a layer containing an instance of an element detail and an overview layer). This suggests that transparency user interfaces are useful.

Three-dimensional scene creation is of interest for many researchers at the human computer interaction paradigm and design research. For example, SKETCH is a system (Zeleznik, Herndon et al. 1996) that enables users to use gesture commands to specify the creation of 3D objects. Sketching three axial arrow lines will generate a box with corresponding dimensions. The efforts from Grimstead and Martin describe the method of constructing a solid model from a 2D line drawing. The process includes hidden line removal, line labelling, region recognition and alignment, and adjustment of vertices (Grimstead and Martin 1995). Digital Clay has a similar approach, however, unlike Grimstead's and Martin's approach, it uses freehand drawing as a front end interface instead of line drawing. It is a 3D isometric sketching environment that uses constraint propagation method of Huffman and Clowes (Schweikardt and Gross 1998) to infer the occluding, convex and concave edges to build 3D model.

Quick-sketch (Eggli, Bruderlin et al. 1995) automatically adjusts angles and connects freehand sketches to infer construction of geometric shapes and arcs in a 2D view. It also extrudes the plan profile to generate 3D shapes using object library from a graphical user interface toolkit. The project

DDDoolz allows user to add or delete connected voxel boxes in six directions (of the cube) to create 3D scenes (Achten, Vires et al. 2000). Teddy (Igarashi, Matsuoka et al. 1999) enables user to create spherical object models by drawing a 2D profile (a closed oval) then rotate the drawing sideways to draw the section for extrusion.

Our project, VR Sketchpad is well located in this series of research. It uses freehand sketching as a way to create 3D objects. On the most basic level, similar to Quick-sketch, VR Sketchpad uses extrusions to generate shapes. Beyond simple extrusion, VR Sketchpad also supports generations of solid objects such as boxes and spheres, as well as object placements such as furniture layout creation from 2D symbol configurations.

3. VR SKETCHPAD IN ACTION

VR Sketchpad employed Electronic Cocktail Napkin's various capabilities of diagram parsing and customised user training (Gross 1994; Gross 1996; Gross and Do 1996) as a base graphic engine. The symbolic processor in the Napkin program enables designers to train and define their own graphic symbols and gestures (circle, line, arrow, etc) by drawing examples into the system (with a pen and tablet). Contrary to the approach of PDA (Personal Digital Assistant such as Palm Pilot) that can recognise only a fixed set of shapes, our system lets designers create their symbol library with personalised definitions and drawing features (sequence, pressure, angle, etc). VR Sketchpad then provides meaning association (circle to column, arrow to viewpoint) and performs geometry translations. Figure 1 below illustrates the system architecture of VR Sketchpad.

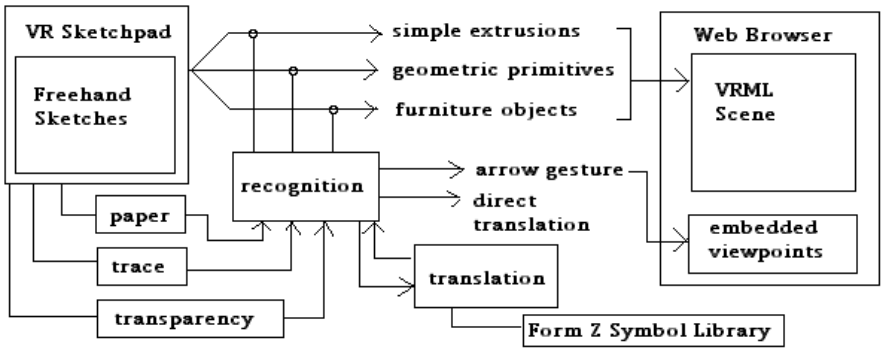

Figure 1. System architecture of VR Sketchpad

Three types of interfaces are available for the sketching input. First, the regular drawing board, paper like interface that allows designers to sketch onto a digitising tablet using a pen stylus. The drawing surface is opaque and displays sketches as drawn. The second type of interaction allows bringing in a picture underlay (raster image) and multiple translucent layers on top. Figure 2 shows a construction drawing brought in to the paper drawing board environment. Several trace layers are overlaid on top to add modifications and annotation diagrams. The thumbnail images on the top portion of the drawing board ("post-it" layer) allows easy management (hide, overlay) of the different trace layers. Each trace layer adds opaqueness to the drawing board while underlay drawings remain visible through the layers. Figure 3 shows a hand drawn diagram on the transparent window overlays on top of other applications (e.g., Form•Z model, screen capture of a drafting document, and a PDF file). The transparent window maintains the drawing functionality of the previous two types of interface (paper, trace layer).

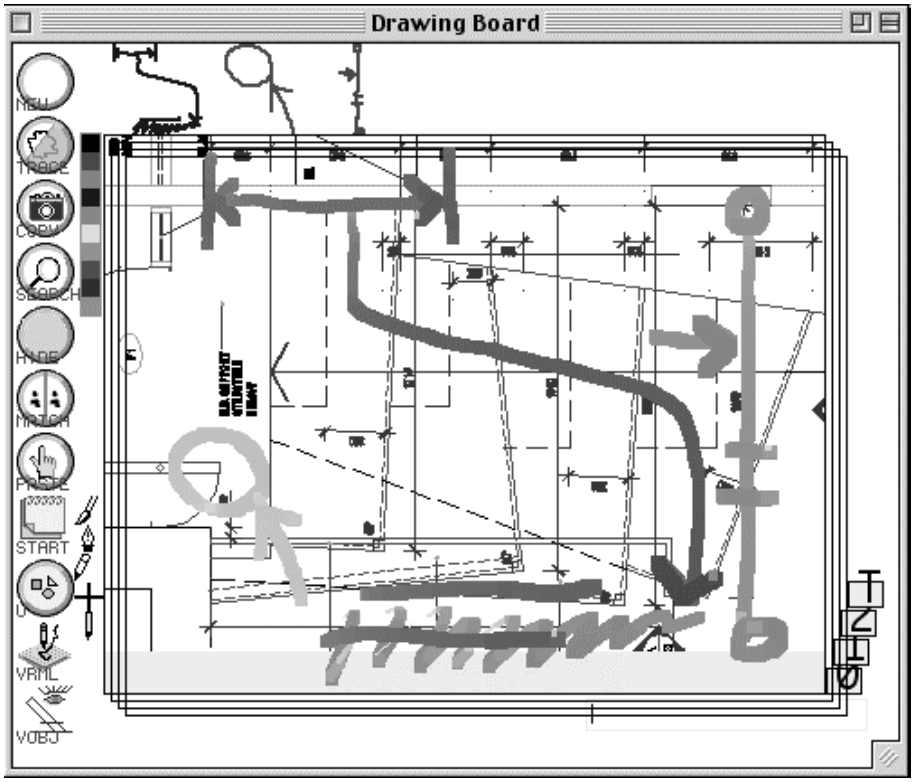

Figure 2. A construction drawing brought in as picture underlay with annotations and modifications on different trace layers. (lower right corner shows tabs for each trace layer for easy selection).

A transparent window is a window (drawing surface) that displays the screen's pixel map as a background for its contents. It appears as though you are drawing on top of the windows belonging to other software applications, or the operating system. With inter-application communication protocols, the Sketchpad would be able to directly interact with the underlay applications. Currently, when user resizes or moves the window, the transparent window instance captures the screen pixel map underneath, and builds the bitmap information into a memory cache. The content drawing method of the window draws the background images first then displays the current user sketch objects.

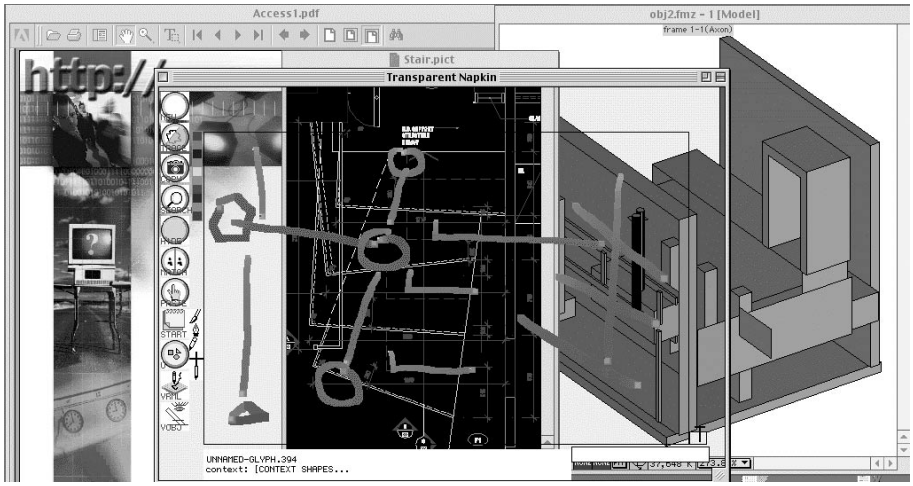

Figure 3. Transparent window overlays on top of three different applications (left to right: PDF document, drafting document, and a Form•Z model). User can sketch and trace images at the transparent window.

With initial input of sketches, VR Sketchpad's processor recognises the drawings (Figure 4 left), it translates the drawing into VRML objects, and then launches a Web browser to display the result (Figure 4 right, VRML enabled Netscape browser).

VR Sketchpad has three levels of diagram recognition and translation processing. First, simple shapes such as lines and circles are extruded to make walls and columns as shown in Figure 4.

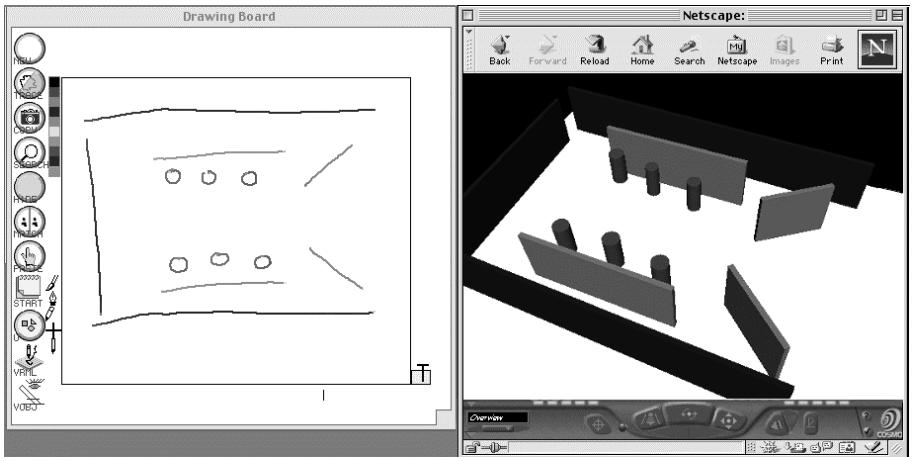

Figure 4. Lines and circles are extruded to make VRML model of walls and columns.

The basic level of translation is simple extrusion. VR Sketchpad also supports extrusions of free form, irregular shapes as desk height (wall height is to the ceiling). The translation processor of VR Sketchpad takes all the user input points in the stroke and converts them into VRML surfaces.

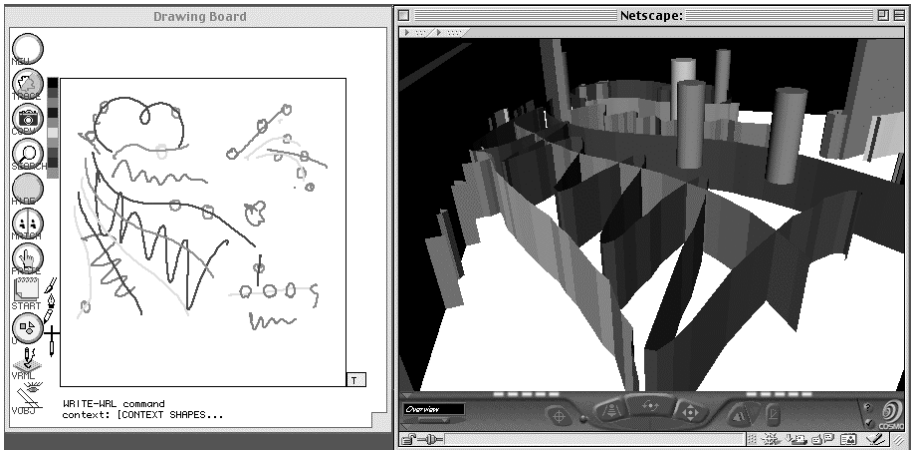

Figure 5. All curve shapes and lines can also be extruded to make curvy partitions. (Note: in this illustration, cylinders are inserted over the partition walls).

The second level of diagram processing deals with solid object mapping and placements. Instead of extrusion, it recognises geometric primitives through symbol conventions or configurations. For example, a rectangle is identified as a box object and translated as a solid cube. A circle that is concentric with a dot indicates a sphere object with desired radius. Figure 6 shows a VRML scene constructed from sketches of geometric primitives.

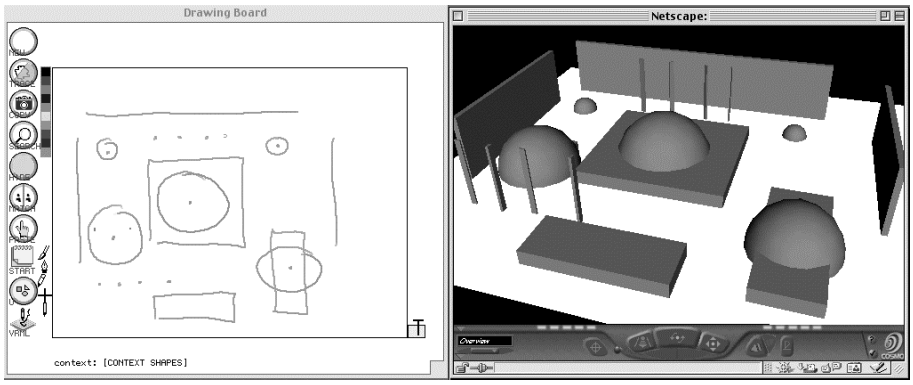

Figure 6. Graphic symbols can be translated into solid VRML object models.

The third level of processing involves recognition of the diagram configurations and the translation with calling of external 3D pre-made objects from a symbol library (e.g., furniture library from Form•Z). Figure 7 shows some shorthand symbols of architecture elements for kitchen and bathroom. Designers can train the configuration processor to recognise graphic standards or personal symbols by drawing freehand diagrams. For example, a toilet can be defined as a rectangle directly above two concentric circles; a dining table set can consist of four chairs (rectangles) surrounding a round table (circle) as shown in Figure 7-9.

Figure 7. Graphic symbols for stove top, sinks, toilet and bath rub.

With the combination of "level one's" partitions (walls and columns) and the translations of higher level complex figures (furniture elements from 3D symbol library), VR Sketchpad user can quickly sketch out an interior layout configuration and see the design in 3D VRML format by pressing a button. Figure 8 shows a furniture layout diagram and the corresponding 3D scene. Notice that all the furniture objects are in real human scale because we use real life object symbols from the 3D graphic library. Diagrams in this instance serve as placement reference instead of scale.

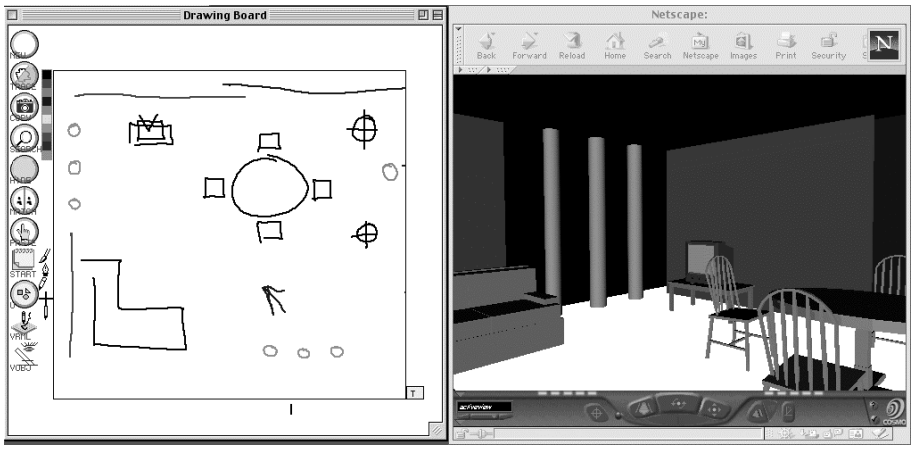

Figure 8. Furniture layout sketch (TV, couch, dining table set, columns, and walls) creates a 3D VRML world.

One important feature of VR Sketchpad is that it also supports gesture recognition for user to specify preferred viewpoints in the 3D VRML scene. The arrow in Figure 8 defines a standpoint and viewing direction toward the scene. The Web browser window on the right shows a particular view angle as indicated on the drawing on the left.

Figure 9 shows that user can draw a sequence of arrows to indicate location of interests and therefore define a viewing path into the 3D world. There is no limit on how many kinds of object configurations or how many objects can be placed in a 3D VRML scene (memory permitting). Users can define their own shorthand symbol sketches and use them to indicate desired element placements. The 3D scene in Figure 9 shows a particular view (lower left arrow on the floor plan) behind the columns looking toward the dining table and chairs, with walls and columns in the background.

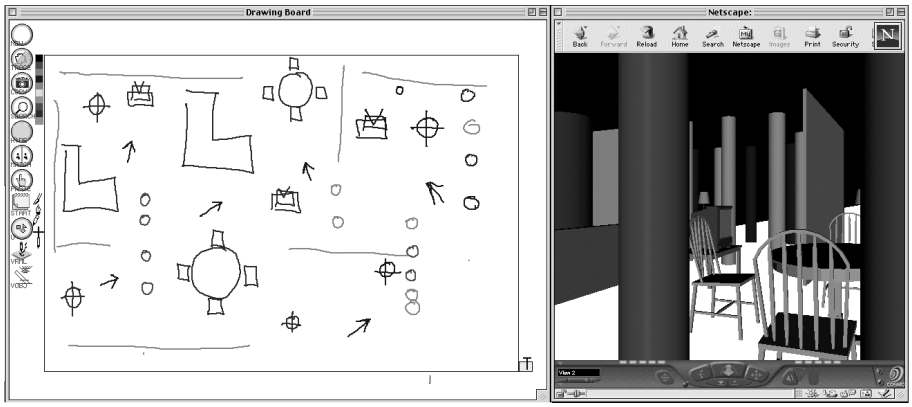

Figure 9. A sequence of arrows in the floor plan sketch indicates places of interest and provides a guided tour to the 3D VRML scene. The bottom left arrow on the drawing (left) shows a location and view into VRML scene on the Web browser (on the right).

4. DISCUSSION AND FUTURE WORK

VR Sketchpad is an application that allows user to sketch in two-dimensional freehand diagrams to quickly generate three-dimensional VRML scenes on the web. The idea of the project came from the teaching of beginning architecture students. The novice architecture students have a hard time to see the relations between the two-dimensional floor plan as a three-dimensional space. The obvious tool for the students to understand the interplay between plan view and its corresponding space is to extrude their 2D drawings into 3D form. Therefore, VR Sketchpad was created to illustrate the relationship between 2D and 3D. VR Sketchpad has then added many features and now supports not just direct one-to-one mapping of object translations but also the higher level complex figure configurations and substitutions (diagram meaning, as well as replacement object symbols). It also shows great potential for on-line gaming and artefact generation for the Web.

The system implementation is straightforward. The interface is simple, yet powerful. Many designers requested to test the system once they saw it in action. We plan to release the prototype VR Sketchpad to architecture students to use in their design studio for conceptual design. We would also like to conduct formal usability studies of the system to find out the problems and desirable functionality.

We are currently adding a slider function in the drawing palette to allow user to specify the extrusion heights. We are exploring adding a 2D sectional view with the 2D floor plan view. We are extending VR Sketchpad to support creating virtual terrain with freehand sketches. Another direction of

the project is to provide an interface to support the display and selection of product catalogue (e.g., Sweets and Steelcase) items with hand drawn symbols (wall, furniture, etc.). We would like to add VR Sketchpad as an interface to a modelling program (e.g., autodessys inc. recently announced its plan for providing interface into their Form•Z software). We have chosen to implement in VRML format because of its ISO (International Standards Organisation) status and because all modelling programs have translation modules to read it as input.

There are many questions worth investigating. Should we extend 2D sketch with isometric view extractions? A previous project Digital Clay (Schweikardt and Gross 1998) from our research group addresses exactly this concern. What does it mean to be able to sketch in 3D? Shall we sketch into the VRML scene to modify the design "right on the spot"? A project in our research group called Space Pen (Jung 2001) is currently exploring this aspect of sketching interactions in 3D.

VR Sketchpad works, but it also has flaws. For example, the furniture placement function uses only the sketch locations as reference for placement and discards the dimensional info (because the furniture models are constrained to real dimension). We have explored placing element dimensions according to user's sketches. However, this would generate scaled down (or up) furniture (that's not usable in real life). It seems to be a logical next step to embed constraints into the sketches such as Stretch-a-Sketch (Gross 1994) so that user, while drawing, will be aware of the human scale concerns and ergonomic dimensions. Or one could display scale and a grid underlay in the drawing surface. We also plan to extend the transparent windows as an interface to other applications. For example, sketching over a VRML scene on a Web browser could potentially add new elements to the 3D scene.

5. ACKNOWLEDGEMENTS

This research was supported in part by the National Science Foundation under Grant numbers IIS-96-19856 and IIS-0096138. The views contained in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

6. REFERENCES

Achten, H., B. D. Vires, et al., 2000. *DDDOOLZ*. CAADRIA 2000. B.-K. Tan, M. Tan and Y.-C. Wong. Singapore, National University of Singapore: 451-460.

Bier, E. A., M. C. Stone, et al., 1993, "Toolglass and Magic Lenses: The See-Through Interface", *Computer Graphics* (SIGGRAPH '93 conference proceedings), 73-80.

Cox, D. A., J. S. Chugh, et al., 1998. *The Usability of Transparent Overview Layers*. CHI 98: 301-302.

Eggli, L., B. D. Bruderlin, et al., 1995. *Sketching as a Solid Modeling Tool*. Third Symposium on Solid Modeling and Applications. C. Hoffmann and J. Rossignac. Salt Lake City, ACM: 313-321.

Grimstead, I. J. and R. R. Martin, 1995. *Creating Solid Models from Single 2D Sketches*. Third Symposium on Solid Modeling and Applications. C. Hoffmann and J. Rossignac. Salt Lake City, ACM: 323-337.

Gross, M. D., 1994. *The Fat Pencil, the Cocktail Napkin, and the Slide Library*. ACADIA '94. M. Fraser and A. Harfmann. St. Louis, MO: 103-113.

Gross, M. D., 1994. *Stretch-A-Sketch, a Dynamic Diagrammer*. Proceedings of the IEEE Symposium on Visual Languages '94. A. Ambler, IEEE Press: 232-238.

Gross, M. D., 1996, "The Electronic Cocktail Napkin - working with diagrams", *Design Studies* 17 (1), 53-69.

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.

Harrison, B., 1996, *Design and Evaluation of Transparent User Interfaces*. PhD: Toronto, University of Toronto.

Igarashi, T., S. Matsuoka, et al., 1999, "Teddy: a sketching interface for 3D freeform design", *Proceedings of the SIGGRAPH 1999 annual conference on Computer graphics* , 409-416.

Jung, T., 2001. *Space Pen: annotation and sketching on 3D models on the Internet*. CAAD Futures 2001. Eindhoven, Kluwer Academic Publishers.

Kramer, A., 1994. *Translucent Patches - dissolving windows.* ACM Symposium on User Interface Software and Technology, Marina del Rey, CA, ACM Press.

Perlin, k. and D. Fox, 1993. *PAD: an alternative approach to the computer interface*. SIGGRAPH. Anahelm: 57-62.

Schweikardt, E. and M. D. Gross, 1998. *Digital Clay: Deriving Digital Models from Freehand Sketches*. Digital Design Studios: Do Computers Make A Difference? ACADIA 98. T. Seebohm and S. V. Wyk. Quebec City, Canada, ACADIA: Association for Computer-Aided Design in Architecture: 202-211.

Trinder, M., 1999. *The Computer's Role in Sketch Design: A Transparent Sketching Medium*. Computers in Building; Proceedings of the CAAD Futures '99 Conference. G. Augenbroe and C. Eastman, Kluwer Academic Publishers: 227-244.

Zeleznik, R. C., K. P. Herndon, et al., 1996, "Sketch: An Interface for Sketching 3D Scenes", *SIGGRAPH '96* , 163-170.