3D Animation Creation using Space Canvases for Free-hand Drawing

Abstract

In this paper we present a novel 3D animation system using a set of easily manipulable space canvases that support free-hand drawing. Our aim is to support the traditional free-hand drawing while improving the functionality by imitating the 3D animation in terms of free-viewing and free animation. The system design emphasis is on the feeling of “what you see is what you get”.

In our system a user is allowed to create planar and curved canvases and place them in 3D space with six degrees of freedom. Free-hand strokes are drawn on each canvas by using any of a user’s favorite input device; a mouse, a digital pen, etc. Various canvases organized in space form a scene and the animation key frames of those canvases and strokes can be recorded hierarchically to represent different levels of movement in the scene. The camera’s movement is recorded when a user tunes the viewing parameters and chooses to add a key frame in the time line for the camera.

The system is both intuitive by utilizing the advantage of 2D free-hand drawing and has the capability of 3D manipulation of strokes, canvases and a camera without requiring the user to have knowledge of 3D graphics and animation. We demonstrate the usability and efficiency of our system by describing the creation process of several short animation movies.

1 Introduction

Even though the 3D computer-generated animation has been very popular recently, traditional free-hand drawing provides a more intuitive manipulation as well as better artistic result. Therefore in the area of 3D animation, it is still true that many artists and designers retain a habit of working with free-hand when they need to form their idea. For example free-hand work is used for the storyboarding process in film production to give a better idea of how the scene will look and feel with motion and timing.

Existing animation storyboarding systems have been quite successful either in 2D free-hand drawing situations or in 3D environment settings. This allows the Animators and Directors to work out any screenplay, camera positioning, shot list and timing issues that may exist with the current storyboard. Improvement of the storyboard is achieved iteratively towards perfection by creating a new animation each time after the Director reviews the storyboard and provides feedback. Since animation is usually an expensive process, editing the animation at the storyboarding stage helps minimize "deleted scenes" in the film and thus drastically decreases the budget.

Figure 1: A scene created in our system by an animation student using free-hand drawing.

The 2D storyboarding technique [Cri07] (with or without the assistance of a computer) is widely utilized in the animation production process, which most animators are familiar with. The cognitive processes of the artist [Las80; Lim03] can be expressed clearly through free-hand drawing with pencil and paper, which is the handiest way for an artist to present his/her idea and communicate with others. Comparably current 3D techniques equipped with 3D object and camera motions is still less popular due to the fact it requires of a familiarity with 3D space and a good patience in tuning the tedious settings and parameters. Though some 3D storyboarding systems provide quite a large library of existing 3D objects, they still lack of intuitiveness and flexibility in representing the arbitrary real 3D world.

From the observation of 3D animation production, we question ourselves why the free-hand animation has been less and less popular and constantly replaced by 3D animation while many artists still want to use the free-hand version which is also more artistic and more expressive. We consider one of biggest reasons is the convenience of 3D manipulation and free-view visualization of 3D animation. Therefore we take an approach by starting from 2D drawing to the direction of 3D animation by helping the traditional designers to keep their skills with an intuitive 3D interface. We believe that the traditional art production method must be retained and seek certain help from computers instead of replacing them by pure 3D animation.

In this paper we present a system that uses a space canvas and its associated strokes as the foundation for a representation of a scene. Our system provides the user the capabilities to organize the drawn primitives in 3D space and in temporal sequence, while still retaining the essence of a brush and canvas-based sketching manner in order to utilize a user’s strong 2D drawing skills. Some view related issues of our representation of the scene are also considered in our approach.

The rest of the paper is organized as follows. We begin with a discussion on how our system relates to previous work. We then give an overview of our system and describe the implementation of the main features. We then go on to demonstrate the primary

uses of the system: creating scene and animation using our system. We conclude with future directions in this research, based on the experience of animation art students using the system.

2 Related Work

Started from the work to automatically transfer paper-based engineering plans into 3D geometry in the 1960s [CPC04], the research area in sketch based modeling has been greatly progressed by many results in transferring the stroke input into a 3D model [IMT99; IH03; DML04; SWS05; KH06; WL07]. However, the goal of our research is not about how to transfer 2D free-hand drawing into a 3D model, but more about how to increase the flexibility and extend the capability of the free-hand drawing system while still retaining the essence of pen and paper based drawing. Thus, our review focuses on previous research work that uses the strokes or paths directly without trying to convert them to 3D models, while enhancing 2D design productivity and flexibility by means of introducing 3D capabilities.

Some of the research goes into the direction to extend the 2D sketching to allow creation of 3D curves. Cohen et al. [CMZ99] introduce a system where a 3D curve is defined by its first drawing on its screen plane projection and then by its shadow on the floor plane.

Tractus [SS05] supports a drawing canvas to do vertical movement, which makes the device capable of maintaining direct spatial mapping between the real and virtual spaces. The method demonstrates how it permits the drawing of non-planar curvature is remarkable. When drawing with a pen on the canvas surface and moving the canvas vertically, complex 3D paths can be constructed by the user, which is a difficult process with 2D methods.

There is evident appeal by the designers to add the ability to render a single sketch from multiple viewpoints, since the work of the artist can be significantly reduced. In view of the above requirement, some research work has been conducted in this direction.

Tolba et al. [TDM99] use projective 2D strokes in an architectural context. By projecting 2D points onto a sphere, 3D reprojection is achieved under different viewing conditions. It is obvious that the camera position of the scene is fixed at the center of the sphere. Thus, the system is only suitable for creating panoramic sketches, but not yet for the full 3D version with 6 DOF.

Bourguignon et al. [BCD01] use a method to project strokes to a screen aligned plane for a representation of the scene without reconstruction of a 3D model. In their system, to account for view-dependency, the strokes are rendered as thick strokes that may deform and disappear progressively as the camera moves away from the original viewpoint.

A system called 3D6B editor [Kal05] was based on an idea to project strokes onto a user-definable 3D grid. The data that represents the created scene stored as 3D line segments (called strokes). The rendering of the scene is done with a line-based renderer in this system. With the functionality of moving the canvas while drawing strokes, it is able to create non-planar 3D strokes in their system. We found it very useful in creating conceptual models. However, as the number of 3D lines grows when representing a complex scene, it becomes very confusing and hard to control for designers. The inconvenience of manipulation of the canvases is another problem of this system that limits its usability. In addition, since no region-based occlusion is considered in this system, it is hardly applicable for a system where spatial layout reviewing is a major requirement, such as the 3D storyboarding system.

By examining the 3D6B editor [Kal05], Dorsey et al. [DXS*07] realized its difficulty in organizing canvases and its lack of post-creation transformation or alteration of strokes, which makes the creation of a 3D sketch a tedious task. In order to overcome the deficiencies, they presented a system called mental canvas that uses strokes and planar “canvases” as basic primitives with the basic mode of input being traditional 2D drawing. The system allowed a designer to transfer strokes between canvases. They also introduced methods for a user to control stroke visibility. In addition, they provided a few built-in 3D assemblies of canvases in common arrangements: axial cross-sections, parallel stacks, and a circumferential ring.

The above systems have achieved some functionality for 3D free-hand drawing by either drawing 3D curves or drawing 2D curves on reorganizable canvases. However, the shortcomings are also obvious. All of the above systems deal with simple lines that can carry a few stroke properties thus making it less comfortable for artists to be artistically creative. An artistic stroke contains rich properties, such as variant width, texture and transparency. In next section, we will describe our system that is designed for animation prototyping, where view dependent and animation aspects are also considered.

3 System Overview

Our aim is to provide an animation tool that is convenient for artists who prefer free-hand drawing in order to create 2D like artistic works. Our system extends the 2D artistic sketching to 3D, while striving to preserve the degree of expression, imagination, and simplicity of 2D drawing. In a short time our system allows a user to draw on malleable canvases in 3D space. An object is defined as a stroke / a group of strokes / a canvas / a group of canvases. Our system also supports some animation functionality which allows the objects and camera to move in 3D space.

While sharing some basic idea, our system defers from the above mentioned systems, especially the 3D6B editor [Kal05] and mental canvas [DXS*07], in the following aspects:

- **Stroke representation:** Strokes in our system are represented as triangle strips rather than curved lines as with the other systems. Our representation of strokes is more suitable for carrying various artistic properties.
- **Region occlusion:** Since a stroke is defined as a triangle strip in our system, users can easily create a region by overlapping wide strokes with various transparencies**.** All of the above mentioned systems, except the mental canvas, overlooked this functionality and therefore may cause visual confusion when viewing a complex scene. The mental canvas system used a rasterized occlusion map for region occlusion that would suffer from resolution problems when viewing the content on a canvas from a close distance.
- **Supported canvases:** Both planar canvases and curved canvases are supported in our system. The system also supports user-definable polyhedral canvases to allow designers to choose from for some common arrangements. Furthermore, a user is allowed to create a group of canvases, to bind them together to create a canvas assembly, and to export them for future use.
- **Animation:** Both object and camera movements are supported in our system. Users can manipulate the position and orientation of each stroke or canvas, with a full suite of tools

very similar to traditional CAD programs. The position and orientation of the camera are implicitly defined by the interactive view changes; therefore the camera is usually not explicitly displayed in our system.

- **Other features.** Several view related aspects, such as billboarding, triangle simplification, and fogging effect are also considered in our system.

3.1 Stoke Representation

Rather than represent strokes as curved lines, we choose to further convert the curved lines into triangle strips with the lines represented by their center axes, similar to that was defined in [Her98], as shown in Figure 2, where the width of each stroke is a user definable parameter.

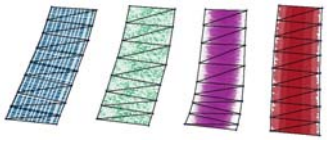


Figure 2: Strokes represented as triangle strips with transparent texture map.

Once the 2D point sequence is entered via a mouse or a graphic input tablet, it is first projected onto the previously selected canvas to generate a curve on that canvas. Two parallel curves on both sides of the projected curve are then generated according to the width parameter either specified previously by the user or obtained according to the pressure value if the tablet pen is used. Both curves are subdivided into line segments and the vertices of the segments are then connected back and forth to create a triangle strip, while texture coordinates for these vertices are automatically generated during this period.

Although simply defined, designers are already able to create artistic graphics using these strokes just by changing the provided parameters: width, texture and transparency, as shown in Figure 3.

Figure 3: Various 2D drawings created using our strokes.

All the strokes can be transformed within canvases after creation. For example, they can be translated or scaled along the two axes of the canvas and the rotation is about the axis perpendicular to the canvas with a given center.

Our system supports stroke grouping, copying and pasting. Once grouped, all strokes are transformed together, and their relative positions are preserved. As a comparison, in mental canvas system [DXS*07], grouping functionality is only supported in canvas level. Thus makes it less efficient to reuse strokes within canvas.

Strokes are allowed to intersect each other so as to give enough freedom for designers to draw arbitrary artistic works. Obviously, when the strokes are drawn on a same planar canvas, all the generated vertices lie in the same plane. This may cause serious aliasing problem at intersectional regions, as we can see from the left side of Figure 4. The system can automatically offset the top stroke vertices at the intersectional region a little bit above to eliminate this artifact. Surprisingly, in our user experiments, some designers would like to keep this artifact as a special effect for design. Thus, we provide this intersection offset mode as an optional functionality.

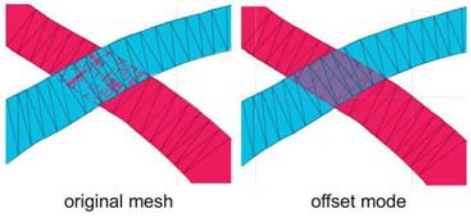


Figure 4: Aliasing when strokes intersect and the solution.

3.2 Region Occlusion and Stroke Visibility

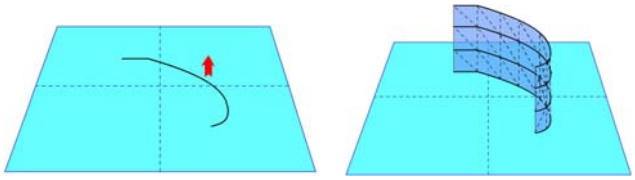
Since each stroke can have an associated transparency, it is free for the designers to choose either to blend the current stroke with the previously drawn strokes or to paint on top of them in an overlay mode.

In either case, when the designer keeps drawing strokes in one particular region, especially when a wide stroke has been chosen, he/she may produce a region filled with strokes that can occlude the behind strokes. As our stroke is represented as a triangle strip it is comparatively easier to create a filled region than in the other systems, by providing a large width for the stroke. In order to improve system performance, we provide a functionality to reduce the number of triangles in this region by first calculating an outline of the region, removing the original triangles and then re-triangulating the outline area. The region occlusion problem is solved in the way that the designer is able to avoid visual confusion when viewing a complex scene.

The view dependent opacity for strokes and one-sidedness for canvas is also useful as suggested in mental canvas system [DXS*07] to help deal with the stroke visibility issue.

3.3 Canvas Creation and Object Management

Canvas management is a basic functionality in our system to extend the 2D sketching to 3D. In our system, both planar and non-planar canvases are supported. Planar canvases are represented by their positions and orientations, and optional parameters are also provided such as texture, one-sidedness, and billboard information. There are two types of non-planar canvases. The first type is generated from a user-specified stroke. Once a stroke curve is drawn on a canvas and a width of the non-planar canvas is given by a subsequent mouse movement away (may or may not be perpendicular to the canvas) from the canvas, the stroke curve will be first subdivided into line segments with a pre-defined constant interval. The width is divided in the same way as the stroke curve has a better ratio between the edges of the triangles to be generated, as shown in Figure 5. To avoid narrow canvases, a width under a certain threshold is rejected; in our system this value is set to be three times of the pre-defined constant interval.



(a) Stroke drawn on canvas (b) Created curved canvas

Figure 5: Creation of a curved canvas.

The second type of non-planar canvas is the polyhedral canvas, created by grouping a set of planar canvases or importing from any simple-shaped polyhedral object.

As we can see from the above description, both non-planar canvases are based on planar basis, thus the non-planar canvases does not increase the complexity of canvas definition.

Similar to strokes, canvases can also be grouped and ungrouped. Canvases, along with their associated strokes in a same group, will be transformed together.

Once a canvas is created in the system, it can be transformed in a similar way as any CAD programs. For example, the user can transform canvases by moving along or rotate about their local coordinate axes, optionally using a single-axis constraint along any of the three coordinate axes. We provide interface widgets also similar to a CAD program to help perform these transformations.

In our system there are different levels of objects that require manipulation, such as a stroke, a stroke group, a canvas, and a canvas group. To avoid confusion when editing a complex scene, we require a user to select an active object for editing one at a time. He/she is allowed to switch between canvas selection and stroke selection modes. By default, the selection is performed upon groups. A fold/unfold operation is provided to toggle selection mode between group and group members, by double clicking the active object for instance. Ungrouped objects are considered as one-member groups and could be selected at the group selection mode. The feedback from the participating designers on the efficiency of this object management method has been extremely positive.

3.4 Stroke-canvas intersection

Once an active canvas is selected, a user can sketch on the 2D screen, which evokes the system to calculate the projective strokes on the active canvas. Since both the planar and the non-planar canvases are finally defined by one or more planes, the projection of a stroke to a canvas is always performed by calculating the intersection between a plane/planes and a ray shooting from each 2D point of the curve drawn on screen following view direction.

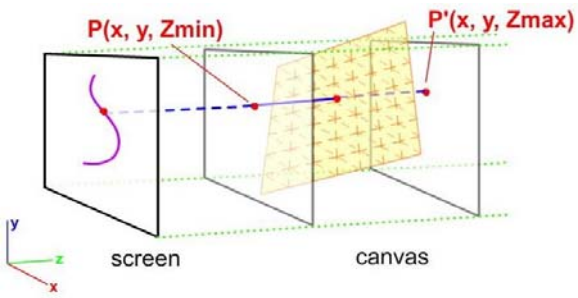


Figure 6: Projecting a stroke onto a canvas.

As an implementation, we use *(x, y, Zmin)* and *(x, y, Zmax)* as the parameters successively, where *(x, y)* are the 2D screen coordinates of the point and *Zmin = -1* and *Zmax = 1* are the two depth values, as shown in Figure 6. A ray connecting the two points *(x, y, Zmin)* and *(x, y, Zmax)* is intersecting with the canvas plane(s) to obtain a projective point on that canvas.

A planar canvas is considered to have infinite size, so that any intersection of the ray and the plane can be accepted as a resulted projection. A bounding-box for the selection purpose is regenerated once the stroke has been projected out of the extent.

A non-planar canvas consists of some planar facets. The intersection is performed first by checking the intersection between the ray and each plane that the planar facet belongs to, and then by checking whether the intersection point is inside the facet. Only the intersection point that lies within a facet will be accepted as a real intersection to be recorded as the projection of the 2D point on the canvas. All facets of a canvas will be traversed until a projection is found. To connect points across two adjacent facets, the intermediate point on the edge between these two facets is calculated to ensure the shortest path to connect these three points. Two successive points that do not lie within one facet or two adjacent facets due to an abrupt move in sketching are rejected to avoid connection problems. Once a projective curve is generated, it is converted into a triangle strip in the way discussed in Section 3.1.

3.5 Animation Creation

An important functionality of our system is that we allow a user to define motions for both objects and camera by recording their transformations at key frames and by interpolating them using a Hermite curve interpolation.

The system also allows a user to define the transformation of objects in a hierarchical way for object groups and group members. The transformations defined for groups are transferred to group members to form concatenated transformations, so that the groups can move together while the group members have their relative movements to the groups. To keep simplicity and clarity, we do not consider the multi-level hierarchy either for objects or transformations in our 3D sketching system.

In order to keep the “what you see is what is get” feeling for a designer/a user, we let a camera not visually appear as a malleable object in the scene like in other 3D animation programs. The transformation of the camera can be obtained implicitly when the user changes the view interactively. When the camera moves around a fixed planar canvas, the sliced appearance of the canvas may be seen at certain angles which may destroy the 3D impression of the whole scene. To address this problem, we allow the user to apply the traditional billboard technique to any objects that are nearly symmetrical and are to be viewed in various angles.

4 Implementation

The system is implemented in Visual Studio .NET 2005. The interface uses MFC and displays using OpenGL. The interface of the system consists of four parts. The first part is the drawing area, where the user can create a canvas, draw strokes and manipulate them. The dialog area contains several buttons and some other interface widgets where the buttons are used to toggle system modes, while the other widgets are used to tune the drawing parameters, such as stroke properties. The menu bar is used to access less frequently used features. The last part is a timeline for setting animation key frames.

There are five major operating modes in our system: canvas creation mode, object selection mode, drawing mode, object transformation mode, and camera mode.

All canvases are created in the canvas creation mode. A planar canvas is created initially on the current view plane, which can be manipulated afterwards. A curved canvas can be created from a base curve drawn on any planar canvas. Polyhedral canvas can be created either by grouping the existing planar canvases, or by importing from a simple polyhedral shape.

In object selection mode, there are two sub-modes: canvas selection mode and stroke selection mode, while both modes can be further divided into group selection mode and group member selection mode, which can be switched. Some object properties can be specified once an object is selected, such as canvas one-sidedness, using billboard or not.

In drawing mode, the user is allowed to draw strokes on an active canvas, which is specified in the previous selection mode. Users can change stroke properties any time in drawing mode.

In object transformation mode, there are also several sub-modes according to object types (a canvas, a canvas group, a stroke, or a stroke group) and operation types (such as translation, rotation, scaling). All object transformations can be specified and stored at key frames. Each object can have its own key frames.

In camera mode, supported camera transformations are similar to those in other CAD programs. Camera key frames can be specified in the way similar to object transformations.

5 Experimental Results

Our system runs on a Windows desktop workstation with a mouse and an optional pressure sensitive input tablet. Our program can be executed at interactive rates, i.e. a designer is able to see the results with changing viewpoints interactively after he/she finishes the drawing. A scene created in our system is rendered exactly the same as the designers have drawn with the selected strokes. There is no change in shading when the view is changed, since no lighting information is applied to the scene. In other words, the scene will remain in the 2D drawing style when it is manipulated. As a result, designers gain a “what you see is what you get” experience when using with our system.

Through several users over several months, our system has obtained feedback that it provides significant advantages over both 2D drawing systems and 3D modeling and animation systems. Compared with 2D drawing systems, our system has the freedom to allow designers to continuously change the viewpoint. As we can see from Figure 7, once the scene is drawn, we can view the scene from different view directions as in any 3D programs, while retaining a sketchy style all the time. Compared with 3D modeling and animation systems, our system keeps the sense of traditional 2D drawing style very well. Designers do not have to follow the tedious 3D modeling, texturing, and lighting process to obtain a final result. The sketchy style of objects provided by our system can hardly be achieved by rendering 3D models in any traditional 3D programs even with some powerful non-photorealistic rendering functionality. Besides, the 2D performance of our system is not limited to the current implementation. Any advanced functionality in 2D painting system could be introduced into our system to improve the drawing quality.

Figure 8 and Figure 9 show another two results created by the animation art students. The quality of the animation produced in our system is quite acceptable. The students were pleased with the results they were able to achieve using our system, with only a shallow learning curve. All of the users were familiar with some 3D modeling programs in order to compare our system with the 3D animation creation process.

6 Conclusion

This paper has presented a novel system for creating 3D animation using space canvases for free-hand drawing. To our knowledge, it is the first system that designers have been able to create a 3D animation in the way almost same as 2D drawing, while enabling 3D capabilities in order to manipulate both the objects and the camera.

The system is able to create rather complicated scenes that consist of only free-hand drawn strokes. All the strokes are drawn in 3D space with the help of either planar or non-planar malleable canvases. Designers are allowed to draw strokes without facing the difficulty to deal with a polygonal mesh or the inflexibility of a parametric pipeline.

Our approach suggests several interesting avenues for future work. The animation of strokes can be more articulated focusing on intuitive user gestures and automatic novel view creations out of existing views can create a larger amount of freedom to go for a full 3D feeling. We also plan to include sound track editing for a complete system.

Acknowledgement

7 References

[BCD01] Bourguignon D., Cani M.-P., Drettakis G.: Drawing for illustration and annotation in 3D. In *Computer Graphics Forum* (sep 2001), Chalmers A., Rhyne T.-M., (Eds.), vol. 20 of *EUROGRAPHICS Conference Proceedings*, EUROGRAPHICS, Blackwell Publishers, pp. C114–C122.

[CMZ*99] Cohen J. M., Markosian L., Zeleznik R. C., Hughes J. F., Barzel R.: An Interface For Sketching 3d Curves. In *Si3d ’99: Proceedings of The 1999 Symposium on Interactive 3d Graphics* (New York, Ny, Usa, 1999), ACM Press, pp. 17–21.

[CPC04] Company P., Piquer A., Contero M.: on the evolution of geometrical reconstruction as a core technology to sketch-based modeling. In *Eurographics Workshop on Sketch-Based Interfaces and Modeling*, August, 2004, Hughes J. F., Jorge J. A., (Eds.), Eurographics.

[DML04] Diehl H., Müller F., Lindemann U.: From raw 3d-sketches to exact CAD product models --concept for an assistant-system. In Eurographics Workshop on Ketch-Based Interfaces and Modeling, August, 2004, Hughes J. F., Jorge J. A., (Eds.), Eurographics.

[DXS*07] Dorsey, J., Xu, S., Smedresman, G., Rushmeier, H., and Mcmillan, L., The mental canvas: a tool for conceptual architectural design and analysis, In: Proceedings of Pacific Graphics, October, 2007.

[Cri07] Cristiano G., Storyboard design course: principles,

practice, and techniques, Barron's Educational Series, October 1, 2007.

[Her98] Hertzmann A., Painterly rendering with curved brush strokes of multiple sizes. SIGGRAPH 98 Conference Proceedings. pp. 453-460. Orlando, Florida. July, 1998.

[IH03] Igarashi T., Hughes J. F., Smooth meshes for sketch-based freeform modeling, ACM Symposium on Interactive 3D Graphics, ACM I3D'03, Monterey, California, April 27-30, 2003, pp.139-142.

[IMT99] Igarashi T., Matsuoka S., Tanaka H.: Teddy: a sketching interface for 3D freeform design. In *Siggraph '99: Proceedings of The 26th Annual Conference on Computer Graphics and Interactive Techniques* (New York, Ny, Usa, 1999), ACM Press/Addison-Wesley Publishing Co., pp. 409–416.

[Kal05] Kallio K. 3D6B Editor: Projective 3D Sketching with line-based rendering. In: Proc. of Eurographics Workshop on Sketch-Based Interfaces and Modeling, pp. 73–79, 2005.

[KH06] Karpenko O. A., Hughes J. F., SmoothSketch: 3D free-form shapes from complex sketches, ACM Transactions on Graphics, 25(3), 2006, pp. 589-598.

[Las80] Laseau P.: *Graphic Thinking for architects and designers*. Van Nostrand Reinhold, New York, Ny, Usa, 1980.

[Lim03] Lim C.-K.: An insight into the freedom of using a pen: pen-based system and pen-and-paper. In Proc. 6th Asian Design International Conference (Oct. 2003).

[SS05] Sharlin E., Sousa M. C.: Drawing in space using the 3d tractus. In 2nd IEEE Workshop on New Directions in 3D User Interfaces (IEEE VR 2005) (Mar. 2005).

[SWS05] Schmidt R., Wyvill B., Sousa M. C., Jorge J. A., Shapeshop: Sketch-based solid modeling with Blobtrees, Eurographics Workshop on Sketch-Based Interfaces and Modeling (2005), pp.

[TDM99] Tolba O., Dorsey J., Mcmillan L.: Sketching with projective 2d strokes. In Uist '99: Proceedings of the 12th Annual ACM Symposium on User Interface Software and Technology (New York, Ny, Usa, 1999), ACM Press, Pp. 149–157.

[WL07] Wang H., and Lee M., Free-form sketch, EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling (2007), pp.53-58.

Figure 7: A glass table and several chairs shown in different view points.

Figure 8: A lady in traditional dress walking in a marble road with a woody surrounding.

Figure 9: A fairy flying across a peaceful lake.