ABSTRACT

The authors have developed software routines to produce 3-D images that can be used with any application. Each graphics application tends to have unique features such as the size of objects being drawn or the coordinate system in which these objects are displayed. For example, one application may display objects such as aircraft in a situational display while another displays aircraft system status information. Clearly, coordinate systems and object sizes will differ between these two applications. The stereographic software techniques described are compatible with any coordinate system or size of object and are also compatible with other graphic transformations such as rotation and translation. Stereographic theory is explained in the context of these software procedures as applied in a study of a 3-D situational display. Ultimately, the combination of this software and respective explanation will enable any user to develop and display 3-D images, reliably placing graphic objects in space.

INTRODUCTION

The mathematical concepts of stereographic (3-D) imaging have been described many times. However, the application of these concepts to image generating platforms has not been adequately documented. This paper begins to fill this void by describing how stereographic theory can be applied to image generation platforms resulting in the accurate and reliable placement of objects in space and supplying software that supports any 3-D stereographic application.

PREREQUISITES

Generating stereographic images has one fundamental requirement: that two unique images be generated, one for each eye. This has been understood for many years as demonstrated by the many forms of stereoscopes used to impart a three-dimensional effect to two photographs or slides. Successfully viewing these images also has requirements. Frequently, left and right eye images are sequentially displayed on a monitor. When using monitors to display the left and right eye images, some mechanism is required to alternately shut off the view from the left and right eye. For example, liquid crystal shuttered glasses have been successfully used. One shutter is opaque preventing view while the other is translucent. Signals to the glasses alternate the states of the shutters controlling the view. Synchronizing the images with a viewing device requires additional electronics that typically triggers off the video signals.

THEORY

This section reviews known relationships between displayed images and the viewer. It also addresses the transformations needed to make use of these concepts in stereographic applications.
Virtual Objects In Real Environments

Displaying a virtual graphic object such that it appears to reside in the same physical space between the viewer and the display or behind the display is directly dependent on parameters defining the viewing environment. These parameters include the display size, the resolution of the display, the distance between the viewer's eyes and the display, and the distance separating the viewer's eyes. It is these parameters that define the viewing environment and dictate the potential stereographic viewing volume.

The first step in determining an object's position relative to either the viewer or the display is to transform the object's application coordinate into the viewer's environmental coordinate system. A graphic image is typically expressed in units that correlate to the domain of the problem being expressed. For example, terrain or an aircraft are typically described in units of meters or feet. Describing objects in units common to the domain of the image being generated provides a more direct and verifiable method to achieve the desired result. Thus, the user should not be required to design his graphic software in units that describe a viewing environment.

We are primarily interested in transforming an object's position along the depth axis from application units to environmental units (inches for this discussion). For simplicity, the depth axis will be referred to as the Z-axis where the viewer is assumed to be looking straight down the Z-axis and the X-axis runs parallel to the bottom of the display. The equation for transforming Z application units to Z environmental units is

\[
Z_{env} = Z_{app} \times \frac{\text{DisplayWidth}}{\text{X_CoordSize}} \times \frac{\text{ViewportUsed}}{\text{ViewportMax}}
\]

Dividing the display width by the width of the application coordinate system determines the number of application units per inch. Multiplying the object's application coordinate by this relation cancels out the application units, leaving the object's Z position described in inches. The remaining relation, size of viewport used over maximum viewport size, determines how many horizontal pixels of the total number of horizontal pictures possible are being used to display the image. Each of these are expressed in units of pixels which are canceled out leaving a unitless ratio. This term accounts for graphic applications using less than the entire display. Figure 1 is presented to help visualize the problem. We see that there are a defined number of application units that can be displayed. The viewport consists of a defined number of pixels and the display has a physical width. Therefore, the number of application units per pixel can be computed and the number of pixels per display inch can be computed. Ultimately, the number of application units per display inch can be computed.

When generating stereographic images, the stereographic effect is used to augment perspective rendering. Perspective rendering causes objects to appear smaller as they translate away from the viewer. For perspective images, the X-Coord-Size does not maintain a constant relationship with the display width. Instead it is a function of an object's position in the depth plane. Figure 2 shows an arbitrary plane being pulled out of the viewing frustum. The viewing frustum is defined by parameters that support the display of perspective images. This arbitrary plane is located between the near viewing plane and the far viewing plane. The application's coordinate system is going to have the fewest application units at the near plane and the most application units at the far plane. Any plane between the near and far planes, defined by an object's Z coordinate, will have a resolution bounded by the near and far planes. The
following equation defines the resolution of the x-axis located at some specific depth position.

\[ X_{\text{CoordSize}} = 2 \cdot \text{Aspect} \cdot Z_{pp} \cdot \tan\left(\frac{\text{FOV}}{2}\right) \]

Assuming a symmetric viewing frustum, its field of view (FOV) can be described as an angle. Taking the tangent of one half of this angle results in a ratio describing the length of the viewing frustum (along the Z-axis) to the width (from the center of the viewing frustum to its side along the X-axis). Multiplying this ratio by a Z-coordinate will return half the number of application units along the X-axis in the plane specified by the Z-coordinate. Multiplying the result by two returns the total number of application units along the X-axis. The Aspect term addresses those viewing frustums that are not symmetric and are basically a ratio describing the relative length of a unit along the Z-axis with respect to a unit along the X-axis. In the second step, the lateral offset needed to generate the stereographic effect can be computed, if given the object's depth described in inches.

Figure 3 illustrates the desired result in computing lateral offsets. The object seen in figure 3.A demonstrates an object (located at point A) viewed behind the display's plane. Points L and R are the required placements of the object for the left and right eye views, respectively. Similarly, figure 3.B demonstrates an object viewed in front of the display's plane. Again, points L and R are the required placements of the object for the left and right eye views. Note that the displacements have reversed when viewing objects behind the display plane as compared to objects viewed in front of the display plane.

The lateral offset is computed using a similar triangle's argument. The triangle seen in figure 3.A composed of line segments AB, AL, and DL is geometrically similar to the triangle composed of line segments AD, AE, and DE.

Line segment BC is one half the distance separating the viewer's eyes. Line segment BD, the distance from the viewer to the display. Both are measurable and thus easily determined. Line segment AD is computed as a result of the transformation described earlier. Therefore, line segment AB equals the sum of line segments AD and BD. Computing the length of line segment DL, the desired lateral offset of the graphic object, can be obtained by the equation

\[ DL = \frac{BC}{AB} \cdot AD \]

As line segments AB, AD, and BC are described in viewing environment coordinates, the lateral offset of the graphic object is also described in viewing environment coordinates. The lateral offset will now have to be transformed back to application units so it is usable with the graphic application.

The final step in causing the stereographic effect is transforming the object's lateral offset from inches back to application units. The equation to perform this transformation is

\[ X_{\text{app}} = X_{\text{env}} \cdot \frac{X_{\text{CoordSize}} \cdot \text{ViewportMax}}{\text{DisplayWidth} \cdot \text{ViewportUsed}} \]

Again, a relationship is used that defines the number of pixels per display inch. Multiplying the lateral offset by this relation cancels the inch units, leaving the offset described in application units. The equation again accounts for the amount of display area actually used.

Limiting The Physical Position Of A Displayed Object

The toolset constrains the volume in which a graphic object can be displayed. More precisely, the toolset limits the depth, how far out of the screen to how far behind the screen, objects can be rendered. These constraints were added due to Vernon's [2] report on limiting
convergence variance. Vernon, et al. provide excellent motivation and support on this topic. Therefore, it will not be discussed in this paper with the exception to say that we have incorporated Vernon's advice.

**APPLIED THEORY**

To simplify the process of creating stereographic images for the Cockpit Integration Division of Wright Laboratory's Flight Dynamics Directorate, a software toolset was developed to handle the transformation of graphic objects from their own coordinate system to graphic objects displayed as stereographic images. The toolset consists of five functions of which only two must the user invoke. Each of these functions applies the theoretical requirements for generating stereographic images as discussed above. The remainder of this section discusses these functions motivating their need and describing the function accomplished.

**Stereo-Initialization**

The function Stereo-Initialization is responsible for establishing the constant parameters of the physical viewing environment. This includes display size, distance separating the viewer and the display, and the distance separating the viewer's eyes. Supporting this function is a configuration file that contains the list of environmental constants that are read by the function. The configuration file was adopted to allow easy adaptability of the display software to various viewing environments. Once the viewing environment parameters have been established this function then computes the maximum and minimum bounds of the stereographic environment.

The Stereo-Initialization function also determines if perspective rendering is to be used by the supported program. As discussed earlier, perspective rendering requires additional calculations in determining the size of the displayed X-axis. This parameter will be used by another function to determine if these calculations need to be performed.

The parameters established by this function are constant and are assumed to be valid for the duration of the executing program. Therefore, this function needs to be called only once by the executing program.

**Stereo-ZtoX-Transform**

The function Stereo-ZtoX-Transform is responsible for computing the lateral offset based on a position in the depth plane. The function accomplishes this in two steps. First, it determines the width of the specific x-axis at the object's z-coordinate. Then it computes the lateral offset to the object's x-coordinate.

This function computes an object's lateral offset by invoking three functions: Application-To-Environment, Compute-X-Offset, and Environment-To-Application. These three functions basically accomplish the transformations described above and are hidden from the user. The details of these functions will be described later.

This function must be called for every graphic object being rendered in the stereographic volume. Depending on the fidelity of the stereographic images being generated this function could be called for every vertex defining an object.

**Application-To-Environment**

The function Application-To-Environment transforms a graphic object's z-coordinate from application units to inches, the unit we used to describe the viewing environment. This calculation is achieved precisely as described earlier. As previously discussed, the Application-To-Environment function then bounds the transformed z-coordinate by the near
and far stereographic viewing planes. The transformed z-coordinate is returned to the Stereo-ZtoX-Transform function.

**Compute-X-Offset**

The function Compute-X-Offset accepts the transformed z-coordinate, now described in environment units (inches) from the Stereo-ZtoX-Transform function. The function computes a lateral offset using the similar triangles computation described earlier. The resultant lateral offset value is then returned to the Stereo-ZtoX-Transform.

**Environment-To-Application**

The function Environment-To-Application accepts a lateral offset described in inches as an input parameter from the Stereo-ZtoX-Transform function. The function transforms the lateral offset from inches (environmental units) to application units. The transformed lateral offset is then returned to the Stereo-ZtoX-Transform function.

**APPLICATION OF TOOLSET**

The toolset was developed for and first used by the Cockpit Integration Division's Advanced Crew-Tailored Cockpit Concepts program. They had a requirement to place graphic objects precisely in a 3-dimensional volume within the cockpit \[1\]. The study concentrated on designating one or more JTIDS-like icons in a 3-D space. Subjects would alternately use one of several tracking mechanisms that would cause a cursor to translate throughout a 3-D volume. Once the cursor was in contact with a graphic object, the subject could then designate the object. The stereographic toolset was used to deterministically cause the JTIDS-like icons to appear at specific depths within a volume. The toolset was also used to cause the cursor to appear at a depth directed by the tracker.

The graphic software was hosted on a Silicon Graphics IRIS 4D320VGXT workstation. The output video signal was sent to a Spectrographics monitor. Liquid crystal shuttered glasses connected to the Spectrographics controller were used to control left and right eye viewing of the respective left and right eye images generated.

Figure 4 shows one of the icons used in this study. This particular image was used for training, helping subjects recognize particular icons used in the study. Note that this figure contains two images. The top image represents the view displayed to the left eye and the bottom image represents the view displayed to the right eye. Inspecting the images, one sees the icon seen in the top image left of center. Likewise, the icon seen in the bottom image is right of center.

The stereographic toolset enabled the Advanced Crew-Tailored Cockpit Concepts program to deterministically place objects at precise and predetermined locations. The software successfully allowed subjects to perceive the various graphic icons at the 5 different depth planes required by the study. Also, the software's rapid reconfigurability allowed the experimenters to vary the near and far stereographic thresholds to accommodate all subjects. The configuration file was rapidly modified between check-out trials eliminating the requirement for recompilation.

**SUMMARY**

This paper discusses a software tool set that supports the precise placement of graphic objects in a stereographic display environment. The tools described were designed to not constrain the user to environmental coordinates but to allow him the use of natural, domain-related coordinates making the toolset useful for any stereographic application. The utility of each tool was described in the context of the theoretical requirements that must be achieved to
produce stereographic images whose graphic objects are precisely placed in the image's depth plane.

REFERENCES


Figure 1. Visualization of the Graphic Application

Figure 2. Viewing Frustum

Figure 3A. Lateral offsets of an object viewed beyond the display plane.

Figure 3B. Lateral offsets of an object viewed in front of the display plane.