The Virtual Display Case

Geoff Wyvill
Computer Science Department
Otago University
PO Box 56
Dunedin, New Zealand
geoff@otago.ac.nz

Tracy Mason
talorza@cs.otago.ac.nz

Garry Downes
downesgr@cs.otago.ac.nz

Mark Williams
mark@williams.co.nz

Abstract

Looking at a picture on a screen is very different from looking at a solid object. But if the picture changes correctly as the observer moves, we can create the illusion that there is a solid object "just behind" the screen. We have built such a display using only a domestic quality video camera and a Macintosh PowerPC computer. The user wears a pair of special "glasses" that can easily be recognised in the image of the tracking camera. The display provides a starting point for a series of experiments in VR and latency issues.

1. Introduction

This research started as an offshoot of a project in design of freeform surfaces [6]. We designed an editor with which the operator manipulates a virtual object appearing on a conventional workstation screen. Even though the operator can rotate the virtual object, we quickly discovered that most people found it very difficult to understand what they were seeing. It seems that rotating an object remotely does not give you the same "feel" that you get from rotating an object in your hand. A sculptor working on a large piece acquires a detailed appreciation of the shape by walking around it. Not only does he or she get multiple views but the mechanics of walking gives additional information about where each view is taken from. We can understand the shape of an object in a museum display case by walking around it, even if we can see it from only one side of the glass.

We experiment with the idea of making our computer display behave like a museum display case. The object is supposed to reside just behind the screen which behaves like the glass front of the case. The observer's position is tracked so that the display can be updated to maintain the illusion.

This, of course, is not a new idea. In particular, the Responsive Workbench project [1] does the same thing with stereoscopic effect using Silicon Graphics' specialised display hardware. Our objective was to see how useful an illusion we could produce using only a standard desktop computer and an ordinary video camera.

2. Hardware

Figure 1. Hardware configuration.

Figure 1 shows our complete apparatus. The computer is a Power Macintosh 7600/120 and the camera is a Panasonic NV-S7A. We are not using the recording function of the camera so an even cheaper model could be used. The Macintosh is able to capture a 768x576 pixel image from the camera at 15 frames per second and our image processing

0-8186-7825-9/97 $10.00 © 1997 IEEE

can cope with that data rate. We have also used the faster Power PC 8500/180 with which we can get 25 frames per second.

The user wears a pair of trackable spectacle frames that we call "the glasses" even though they have no lenses. The ones we are using are made of cardboard and they are decorated with coloured spots. As explained in Section 4, we need to know only the spacing of the dots on the glasses to determine their position in 3D.

3. Finding spots

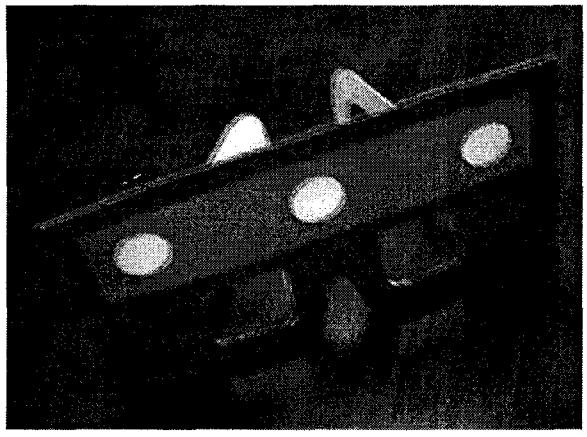

Figure 2. Trackable "glasses".

This is not a sophisticated exercise in pattern recognition but more of an engineer's solution. The glasses are decorated with yellow spots on a blue background. The colours have been chosen to be different from likely colours of clothing or other objects in the camera's view. This, however, is not of major importance. To be recognised as the glasses, an object must have exactly three spots whose centres lie in a straight line in the image plane.

The glasses are found in three stages: firstly the spots are identified, then a set of three spots in a line is recognised as the image of the glasses, and finally the position of the glasses in 3D is determined. We tried using only a grey scale image but it was too easily confused by shadows and a good contrast between spots and background was not always obtainable. We therefore decided to use hue and saturation as discriminators. The raw RGB colours imported from the video were converted to HSV using the standard formula [2]. The V component is the one most affected by shadows. For the purpose of recognition a range of permitted H and S values is defined for each colour.

The whole image is scanned. If a pixel is blue then its neighbours are scanned with a flood-filling type algorithm. If a yellow pixel is found, its neighbours are scanned in a similar way so that an area of yellow pixels surrounded by blue pixels is identified. Each blue area is scanned in this way to see if a complete set of spots can be found. If a complete set has been found the algorithm terminates. Otherwise it continues to look for more areas of blue. A simple average of the positions of pixels making up a spot yields the position of the spot to about 0.5 pixels. Greater accuracy is possible by utilising the colour values of pixels on the edges of the spot [3] [4].

Figure 3. Spot finding algorithm.

For the moment we are using just three spots, enough to locate a point centrally between the eyes and a little above them. By using a four or five spot pattern we could also deduce the complete position and orientation of the glasses. In this paper we deal only with the three spot case.

4. Where is the viewer?

Methods of finding 3D geometry from two or more camera views are well established. See, for example, Wolf[5]. However, we have a subtly different problem. We have a complete description of the object under scrutiny (the glasses), and we want to find where it is in 3D space. This is equivalent to a calibration problem in photogrammetry: locating a camera from known points in the scene. The camera geometry can be approximated as a point within the lens system and a projection plane that is mapped onto the pixel image. We define the *camera coordinate system* as follows:

- The origin is the centre of projection.
- The Z axis is the perpendicular from the origin onto the projection plane.
- The X and Y axes run parallel to the edges of the rectangular image in the projection plane.
- X, Y, Z form a right handed system.

The location algorithm finds the glasses in the camera coordinate system. The three spots that represent the glasses

Proceedings of the Computer Graphics International (CGI '97)
0-8186-7825-9/97 $10.00 © 1997 IEEE

1. Test pixels to find the background (blue). If not found algorithm fails; collect next frame.
2. Start blue_queue with pixel, tag it as seen.
3. While blue_queue is not empty:
    Look at eight neighbouring pixels.
    If pixel is not tagged
        Tag it as seen.
        If pixel is blue put on blue_queue.
        Else if pixel is yellow
            Start yellow_queue.
            While yellow_queue is not empty:
                Look at eight neighbouring pixels.
                If pixel is not tagged
                    Tag pixel as seen.
                    If pixel is yellow, put it on yellow_queue.
                    Else if pixel is blue, put it on blue_queue.
            Calculate dot middle, and increment dot_counter.
4. If dot_counter = number of dots on the glasses then glasses are found and search is ended.

Figure 4. Spot finding algorithm.

are identified as three points in the projection plane. These points define three lines in space; each line joins one spot centre to the origin. The centres of the spots on the glasses in 3D lie on these lines and the distance between the dots is known. In Figure 6, q is the point on the projection plane that represents the centre spot of the glasses and the three lines are labelled $right$, $left$ and $centre$.

We first find a line through q and parallel to the line of spots on the 3D glasses. Assume we have an approximation to this line. Let p be the point where the approximate line crosses $right$. Now let r be the point on the line pq such that the distance pq equals the distance qr. If r lies between the lines $left$ and $centre$ then p is too close to the origin, otherwise it is too far away. This property enables us to find the correct position of p by binary search between $origin$ and a sufficiently distant point along the line $right$.

We prove this property as follows: There is only one point, p, for which r lies exactly on the line $left$. As p approaches $origin$, the line pq becomes parallel to $centre$. When the angle between pq and $centre$ becomes less than the angle between $centre$ and $left$ the point r must lie between $left$ and $centre$. Similarly, we can show that if point p moves far enough away from $origin$, r will lie to the left of $left$.

Once we have found the line pq such that r lies on the line $left$, we can find the position of the three spots in 3D camera space by similar triangles because we know the actual separation of the spots on the glasses.

Compared with the time taken to capture an image and locate the spots, the iteration to fit the spots to the given lines is very fast. Therefore we have used a simple binary search

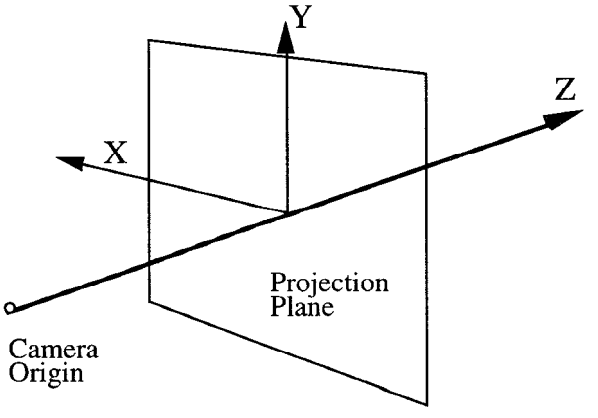

Figure 5. Camera Geometry

Proceedings of the Computer Graphics International (CGI '97)
0-8186-7825-9/97 $10.00 © 1997 IEEE

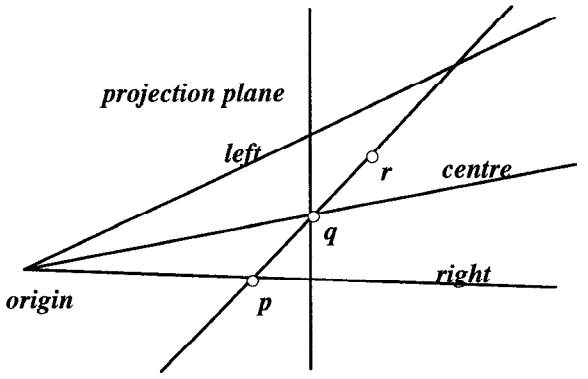

Figure 6. Location algorithm

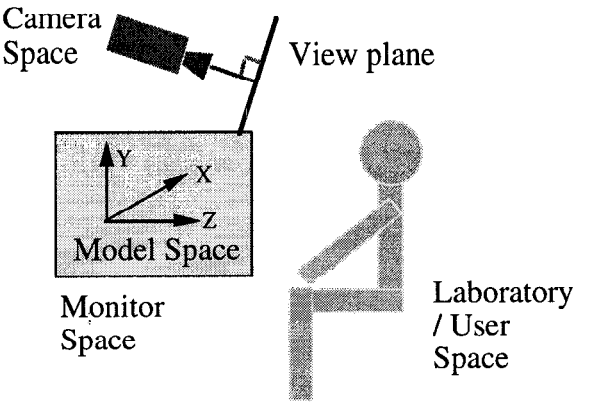

Figure 8. Four coordinate spaces

algorithm that is robust and easy to understand.

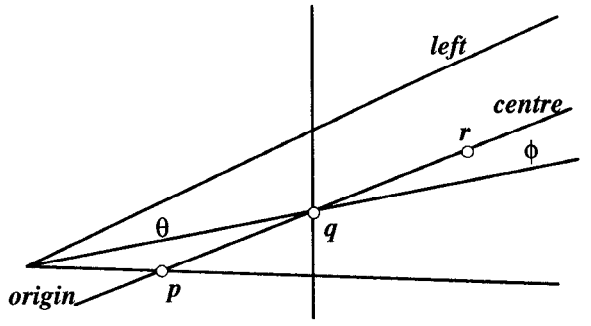

Figure 7. As p approaches the origin angle ϕ becomes less than angle θ and r must lie between *left* and *centre*.

5. Display

To display the model as if it has been seen from the viewer's position we have to relate four coordinate spaces. The model exists in an abstract space in which there is a view plane that we identify with the monitor's screen. The screen's position must be known in the world space, where our operator sits.

The camera space must also be known in the world space. We calibrated the camera angle by the simple procedure of making a picture of a ruler at a measured distance. This gave us a translation from pixel coordinates in the captured image to millimetres in a standard projection plane at one metre.

The centre of the display window is used as the origin of our world space. The camera space is represented by four vectors, c, v, u, t: c is the position of the centre of the camera's lens system and v, u, t, are unit vectors that describe the orientation of the camera where:

v is the viewing direction.
u is the up vector.
t is a vector perpendicular to u and v.

So t, u, v, give the directions in world space of the X, Y, Z axes of the camera space. A point, (x, y, z), in camera space is at the position: $c + xt + yu + zv$ in world space.

The vectors c, v, u, t, can be found by direct measurement. A minor problem was locating the centre of the camera's lens system, see below.

So far, we have used only polygonal models and we are displaying them directly using QuickDraw 3D.

6. Accuracy and latency

The literature of photogrammetry covers elaborate calibration procedures designed to compensate for aberrations in lenses and other sources of error. Certainly these techniques could be applied here although for our purpose such accuracy is not necessary. Automatic calibration of the camera and its position could be done by preparing a board with a fixed pattern drawn on it and making an image of it in a fixed position with repect to the monitor. This would eliminate the need to find the centre of the lens system.

A far more important problem is latency. How long does it take for the system to update the image on the screen in response to a head movement? Even if the processing time is negligible, we should expect a delay of one frame time for the capturing camera, 40ms, plus one frame time for the display, 12ms. Processing time is never zero and our best update time to date has been around 80ms. Since the latency

Proceedings of the Computer Graphics International (CGI '97)
0-8186-7825-9/97 $10.00 © 1997 IEEE

in experimental VR systems is commonly a significant fraction of a second and sometimes several seconds, this is good for a cheap system. But it is not good enough to produce a satisfactory illusion that an object is stationary within the virtual display case.

Our plan is to introduce a predictive element into our system. We do not have to respond to head or eye rotation. Only movement of the whole head changes the viewpoint. And the head cannot accelerate too much without disturbing the visual system to the point where errors would not be noticed. We propose, therefore, to use a simple, constant acceleration model to predict the position of the head and draw our image accordingly.

7. The state of the system

The spots are identified robustly over a range of normal lighting conditions. Bright sunlight entering the laboratory does give us a problem as would placing the camera so that it faces a window. We can locate the spots to about half a pixel and locate the glasses well enough to display the final image from a reasonable looking angle.

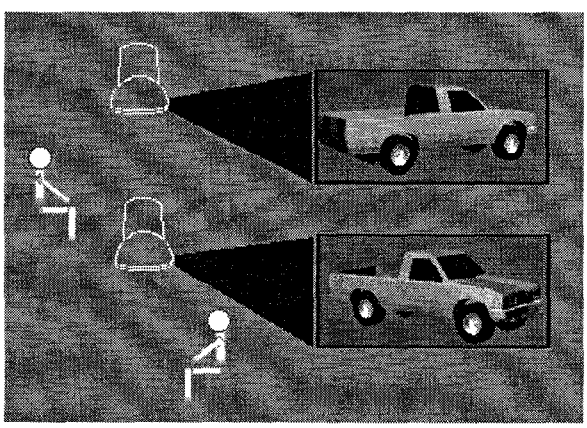

Figure 9. Two views of an object as the eye moves in front of the monitor.

We have achieved a convincing illusion of a stationary 3D object behind the screen, provided that the observer doesn't move too quickly. This will be improved by means of a combination of increased accuracy from sub-pixel position identification and prediction of head movement.

The accuracy of the system has been checked subjectively by sticking a small paper cube to the screen next to the generated image of a cube. As the head is moved the edges of the real and virtual objects can be seen to remain parallel. This test revealed an error in our estimate of the centre of the camera's lens system and we were able to use the test to refine our calibration. An error of one centimetre in the camera position is easily detectable by this test.

8. Discussion

We have demonstrated a display that responds effectively to the motion of an observer and this display requires no more than a typical desktop computer with video capture. We have only just begun to experiment with the system and we do not know how well it will satisfy our original desire to have an easily understood display for designers.

Simulations do not have to be completely realistic. Early flight simulators had very crude displays but were nonetheless adequate for pilot training. If we provide the right cues, an observer may accept the illusion of a 3D object by conscious effort and thereby correctly interpret the shape.

There are many ways to improve the display. Since the pixel processing is fast compared with the image loading, it may be possible to reduce the latency by using a lower resolution camera and recover the accuracy by sub-pixel processing.

References

[1] Stanford University: Responsive workbench. http://www.graphics.stanford.edu/projects/RWB/.

[2] J. Foley, A. van Dam, S. Feiner, and J. Hughes. *Computer Graphics: principles and practice*. Addison Wesley Publishing Company, 2nd edition, 1990.

[3] C. Fraser. State of the art in industrial photogrammetry. *International Archive of Photogrammetry and Remote Sensing*, 25(B5):166–181, 1988.

[4] E. Mikhail, M. Akey, and O. R. Mitchell. Detection and sub-pixel location of photogrammetric targets in digital images. *Photogrammetria*, 39:63–83, 1984.

[5] P. R. Wolf. *Elements of Photogrammetry*. Singapore: McGraw-Hill, 2nd edition, 1983.

[6] G. Wyvill and D. McRobie. Local and global control of cao en surfaces. In *Communicating with Virtual Worlds, Proceedings of CGI'93*, pages 216–227. Springer-Verlag, 1993.

Proceedings of the Computer Graphics International (CGI '97)
0-8186-7825-9/97 $10.00 © 1997 IEEE