Display Memory Access Issues and Anti-Aliasing with a Virtual Reality Graphics Controller.

Matthew Regan* and Ronald Pose*.

Department of Computer Science, Monash University,
Clayton, Victoria 3168, AUSTRALIA.

Abstract.

An address recalculation pipeline used in conjunction with image composition enables the profitable technique of prioritized rendering to be employed in virtual reality applications. These techniques however require the use of large amounts of fast static memory. Aliasing is also introduced in the system and must be removed with special anti-aliasing hardware. The demands on the display memory are quite severe and an unusual memory architecture and addressing scheme are employed to meet those demands. This results in the required performance and excellent memory efficiency while requiring relatively few components. The system provides far better performance at a lower cost than high-end conventional graphics computer systems running virtual reality applications.

CR Descriptors: I.3.1 [**Computer Graphics**]: Hardware Architecture --- *Raster display devices*; I.3.3 [**Computer Graphics**]: Picture/image generation --- *Display algorithms*; I.3.6 [**Computer Graphics**]: Methodology and Techniques;

1.0 Introduction.

The address recalculation pipeline [2] is a new graphics controller which removes the latency to head rotations which plague most virtual reality implementations. When used in conjunction with image composition the address recalculation pipeline may also lead to significant savings in the rendering required to maintain the illusion of immersion within a virtual world [1][4]. The techniques of image composition and prioritized rendering are only useful in the realm of virtual reality due to the independence of the rendered image and the user's head orientation when using an address recalculation pipeline. Prioritized rendering in particular enables reductions in rendering of orders of magnitude and also allows effective exploitation of parallel rendering hardware. This technique is very effective at reducing the rendering required by a translation and therefore the latency to user translations. Fluidity of animation within the world is also improved. The performance achievable by such a system is well beyond that attainable with conventional graphics computer systems, even those at the high end of the market.

*E-mail: {regan,rdp}@cs.monash.edu.au

The cost of a virtual reality system with an address recalculation pipeline is actually considerably lower than trying to achieve equivalent performance with conventional graphics architectures. However the hardware is not trivial and it is important to get the best performance from the investment. Address recalculation pipelines operate on the principle of having a display memory which surrounds the user rather than appearing as a rectangular screen in front of the user. Thus much more memory is required than for a conventional graphics system. As well there is a need for a number of large, fast lookup tables which provide inputs at various stages of the pipeline. The access patterns for the display memory are rather random compared with the linear access patterns for conventional systems. This necessitates the use of large, fast, and relatively expensive static memories for the display memory. To allow for image composition and prioritized rendering one must have multiple display memories, also providing potential parallelism. All in all over 100 Mbytes of static memory chips have been used in our small prototype. However we are able to demonstrate performance which eludes commercial systems costing over a million dollars.

Another consequence of the address recalculation pipeline is the need for hardware anti-aliasing. Since the hardware employs a sampling technique image aliases are introduced. The nature of the processing means that software cannot perform the required anti-aliasing hence the hardware pipeline includes anti-aliasing stages. The anti-aliasing imposes further demands on the display memory systems, in particular requiring a number of adjacent pixels to be processed concurrently.

In this paper, the principles and mechanisms behind the address recalculation pipeline are outlined. The display memory architecture is described and the nature of the aliasing introduced by the sampling nature of the address recalculation pipeline is shown. An anti-aliasing scheme is presented as is an efficient implementation which integrates it with the display memories. The display memory addressing mechanism by which the required anti-aliasing data is obtained is given particular prominence and is one of the keys to the success of the address recalculation pipeline in that it is achievable with relatively few hardware components.

2.0 The Address Recalculation Pipeline.

An address recalculation pipeline is a hardware implemented algorithm which performs viewport orientation mapping after rendering. Rather than using a simple conventional counter for

display memory access the addressing mechanism becomes quite complex and provides a correction for wide angle viewing lenses and user head orientation as pixels are fetched from display memory.

The user head orientation doesn't need to be known accurately until the first pixel of a frame is to be displayed on the output device. As a result the latency to user head rotations caused by computational delays is in the order of two microseconds. This latency is independent of scene complexity and renderer overload.

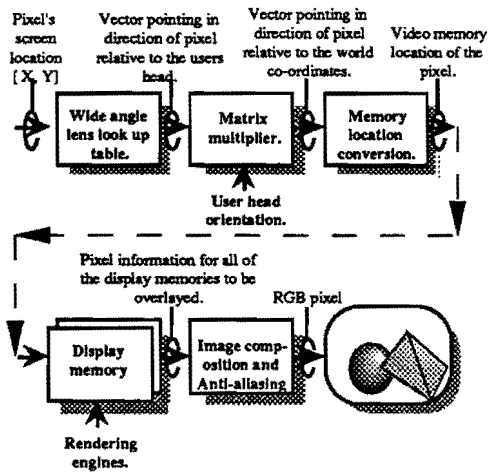

Figure 1: The address recalculation pipeline.

The pipeline is depicted in Figure 1. The first input to the pipeline is the X-Y screen location of the current pixel to be displayed. This screen location is provided by a conventional graphics controller at a normal pixel display rate. A look up table called a Wide Angle Viewing Lens Lookup Table converts the screen X-Y location into a 3-D unit vector pointing in the direction at which the pixel is seen by the user, through the wide angle lenses, relative to the users head. The look up table has one entry per display pixel where each entry consists of three 16-bit vector components. For a display device of resolution 640 by 480 pixels the lookup table will require a memory of 1024(cols)*512(rows)*6(bytes per entry) = 3 Mbytes. Many head mounted displays use wide angle viewing lenses which preserve a standard viewport mapping, however if a special mapping is required for higher fields of view[5][6], the lookup table may be loaded with a new lens mapping, compensating for the lens mapping without run-time penalty.

The 48-bit output of the wide angle viewing lens feeds into a matrix multiplier which forms the next stage of the pipeline. The multiplier multiplies the pixel direction vector with a 3 by 3 matrix containing user head orientation information. The resulting output vector points in the direction at which the pixel is seen by the user, through the wide angle viewing lenses, relative to the world co-ordinate system. The pixel direction vector is fed into the matrix multiplier at pixel display rates while the head orientation matrix is updated at the start of each display frame (ie. after each vertical sync signal). The matrix multiplier is also implemented with 16-bit fixed point arithmetic and is built with nine commercially available 16-bit by 16-bit, 40ns multipliers [8] and six 16-bit, 40ns adders. The output vector from the matrix multiplier is in the form of three 16-bit fixed point vector components [Vx Vy Vz].

The next pipeline stage called the Vector Conversion stage converts the 3D unit vector into a display memory location. The chosen display memory topology for this architecture is the surface of a cube. A spherical topology is also possible [3]. The conversion process involves computing the point at which the ray intersects with the surface of a cube. When the cube is aligned to the axes of the co-ordinate system such that each face of the cube has one of its X, Y or Z co-ordinates fixed at +/- 1.0, the intersection may be computed with a set of parallel divisions with range checks on the outputs of the divisions. For example if the result of the divisions Vx/Vy and Vz/Vy are both within the range (-1.0, 1.0) the ray must intersect with only two of the six faces. The sign of Vy is then used to determine the face of intersection. The point of intersection on the face is then (Vx/Vy, Vz/Vy). The divisions must occur at pixel display rates, so the divisions are performed by a reciprocal lookup followed by a normal multiply using another set of 40ns multipliers. The reciprocal lookup has extra output bits which are used to compensate for classification with fixed precision arithmetic. A programmable logic device is used to accumulate data from the appropriate data paths to multiplex the divider outputs to form the display memory address.

3.0 Hardware Antialiasing.

The address recalculation pipeline performs a remapping of the image in display memory to form an output image in real time based on the wide angle viewing lenses and the user head orientation. This remapping is performed one pixel at a time by the hardware in the address recalculation pipeline. The remapping occurs by sampling the image contained within the display memory and as with any form of discrete sampling, aliasing occurs. Even if the image in the display memory is anti-aliased and rendered with a high quality rendering technique, the hardware sampling occurring will cause aliasing in the final image. The aliases introduced by the address recalculation pipeline cannot be corrected with software in the rendering process. Any pipeline anti-aliasing must occur in hardware.

Providing no hardware anti-aliasing is always an option and in fact simplifies the hardware significantly. Without any anti-aliasing hardware the display image quality suffers quite severely. Images become extremely sensitive to noise from the head tracker. When the intersection computation from the pipeline for a given pixel in display memory is close to the pixel's area boundary, a very small change in the tracked position of the users head caused by tracker noise could cause the output pixel to flicker between adjacent display memory pixels. The flickering will be most noticeable at the boundary of polygons and on straight lines. Even stationary objects will have noticeable artifacts. Straight lines appear broken and the stair casing of angled lines becomes significant.

After simulating the possible artifacts caused by no hardware antialiasing strategy and considering the overall cost of an address recalculation pipeline system, hardware anti-aliasing is considered necessary. Choosing a hardware anti-aliasing strategy may have serious implications on the overall system architecture. The anti-aliasing strategy must be designed with several aims in mind. It must provide an adequately anti-aliasing image, it must be possible to perform the strategy in hardware in real time and it should not dominate system cost or degrade system performance.

The anti-aliasing strategy chosen for this architecture is a linear interpolation filter using redundant addressing bits from the intersection computation. With the current architecture 4 bits in each direction are used for the interpolation filter. A linear interpolation filter provides an adequate trade-off between system expense and filter quality[7].

In order to perform linear interpolation the four pixels surrounding the point of intersection must be fetched simultaneously. The interleaving mechanism for fetching the four adjacent pixels is discussed later.

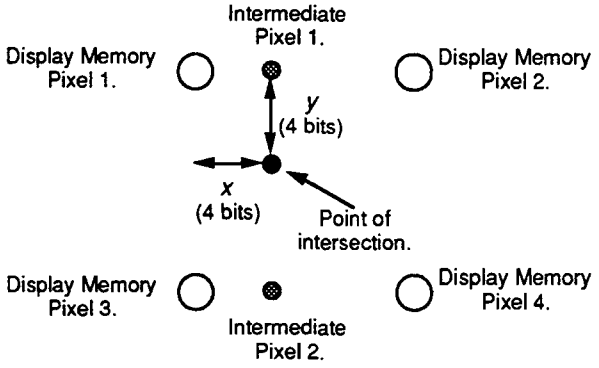

Figure 2. Linear interpolation strategy.

The three color streams of the four adjacent pixels are separated and managed independently. The four adjacent pixels are paired off such that the two Y values are the same. That is pixel 1 is paired with pixel 2 and pixel 3 is paired with pixel 4. Within each pair of pixels, the color component of the first pixel is concatenated with the second pixel color value which is the then concatenated with the (x) redundant bits from the intersection computation. These form two address into two independent lookup tables. The output of the lookup tables form intermediate pixels which are the color interpolation of the two pixel pairs. The intermediate pixel values are concatenated with each other and the (y) redundant bits from the intersection computation, to form an address into another lookup table which interpolates in the X direction. The output of this table is the final color interpolated value of the four adjacent pixels. Figure 3 depicts the overall strategy for a single colour stream.

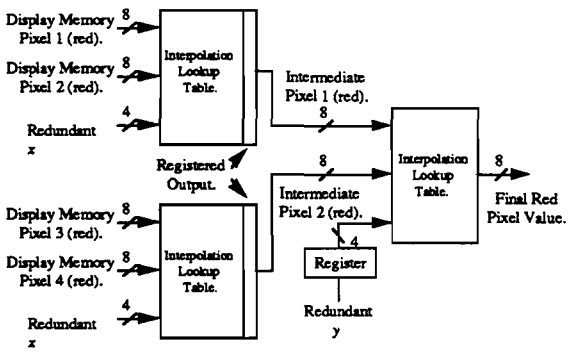

Figure 3. Anti-aliasing lookup table.

The number of bits per pixel color component obviously has a great influence on the size of the lookup tables. When pixels in display memory are stored in a 24-bit color format, each of the three lookup tables for one color streams is 1 Mbyte. The overall anti-aliasing strategy requires 9 Mbytes. As the pixel interpolation occurs after image composition this figure of 9 Mbytes is independent of the number of display memories. While this figure may seem high for an anti-aliasing strategy, a complete system with six display memories[1] would have over 160 Mbytes of display memory and an extra 9 Mbytes is easily justified. When 16 bits per pixel are used for color instead of 24 bits, the overall anti-aliasing strategy requires a little over half a megabyte. The intermediate pixel values are registered hence forming another pipeline stage. The anti-alias lookup table is also required to be static memory operating at video display rates since it forms part of the pipeline.

4.0 Display Memory Access.

The display memory in an address recalculation pipeline system is generally one of the most expensive item in the system, so the organisation of the memory is extremely important, moreover any inefficiencies are multiplied by the use of image composition.

Conventional graphics systems generally use low cost dynamic RAM in a fast page mode or video RAM which has internal shift registers to achieve the high output bandwidth required by raster display systems. The high output bandwidth is achieved in both cases as an artifact of the sequential nature of the display memory accesses. Pixel addresses created by an address recalculation pipeline are not sequential and while there may be some locality in consecutive pixel addresses, these addresses are considered to be random. The display memory must be able to perform random accesses at raster display rates, this means with current technology the display memory has to be implemented with expensive high speed static RAM. Interleaving of the video memory cannot be used to increase the effective video memory access rate which means the random access time of the static ram must be less than the pixel display period, hence making the display memory quite expensive.

The display memory is organised into 6 faces, where each face has a given resolution. The face resolution of our prototype system is 512 by 512 pixels. The faces are arbitrarily labelled in figure 4.

In order to perform a linear interpolation anti aliasing strategy, adjacent pixels within display memory must be accessed simultaneously. The access mechanism used in the prototype system begins by first dividing the display memory into 6 independently addressable segments and then arranging them in such a manner that with appropriate logic, four of the six addressed memory accesses are the required adjacent pixels. The independently addressable segments are labelled A, B, C, D, E and F. The address range of the segments are grouped but different sections of the address range of each segment are grouped differently, for example the address bits 0-7 of segment A are grouped differently to bits 8-17 of segment A. Note that the addressing figures given are for the prototype system with a face resolution of 512 by 512 pixels. The exact grouping is given in Figure 4.

Face intersection computations from the address recalculation pipeline give a 3-bit face number f, and two 9-bit face co-ordinates, i and j. Lets first consider the i face co-ordinate. We arrange the memory such that for a given address the pixel in

segment A in always to the left of the pixel in segment B. The same is also true for segments CD and EF. For a given intersection, if the required order is AB, segments A and B may be accessed with the same addresses, however if the required order is BA, segment A must access the pixel one location further into the memory than segment B. The required order AB or BA is given by the least significant bit of i. The display memory lines are 512 pixels wide which means there must be 256 AB pixel pairs, Thus A and B line position co-ordinates must be 8 bits wide. For any i, the B pixel low order address bits are computed by shifting i right by 1 bit ie. B(0-7) = i>>1. The A pixel low order address are computed by adding one to i then shifting the result right 1 bit. ie. A(0-7) = (i+1)>>1. As A must also be adjacent to C or E. The low order bits of A, C and E are grouped, while the low order bits of B, D and F are grouped.

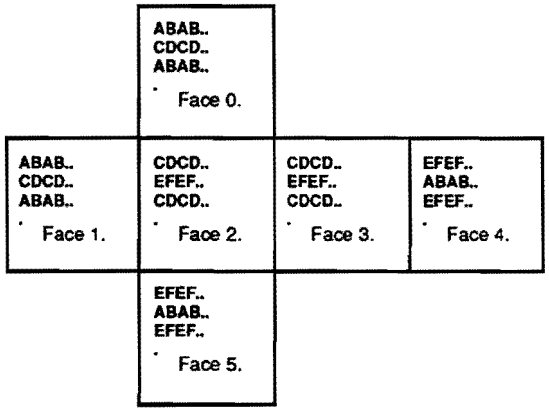

Figure 4. Display Memory and Segment Organisation.

A(0-7) = C(0-7) = E(0-7)
B(0-7) = D(0-7) = F(0-7)

The high order bits of A and B are grouped A(8-17) = B(8-17) as are the high order bits of C and D as well as E and F. Lines of AB may occur in faces 0, 1, 4 and 5. In faces 4 and 5 the lines of AB occur in odd rows, while the in faces 0 and 1 the lines are in even positions. The address bits 8 to 15 of AB may be computed from j, while bits 16 and 17 are computed from f. For odd line positions the address bits 8 to 15 are simply j shifted right one position. When the lines are in even positions the addresses are formed by adding one to j then shifting the result right by one bit. More precisely:

AB(8-15) = j>>1 if face = 4 or 5;
AB(8-15) = (j+1)>>1 if face = 0 or 1;

Similarly the faces of CD and EF are grouped.

CD(8-15) = j>>1 if face = 0 or 1;
CD(8-15) = (j+1)>>1 if face = 2 or 3;
EF(8-15) = j>>1 if face = 2 or 3;
EF(8-15) = (j+1)>>1 if face = 4 or 5;

As stated the two highest order bits of AB, CD and EF (bits 16 and 17) are computed from the face number directly.

The additions required to access adjacent pixels form yet another stage of the address recalculation pipeline and the actual adders are located on the display memory boards. The additions required for adjacent access and the address multiplexing required for double buffering are implemented with a set of four Erasable Programmable Logic Devices (EPLD) per renderer board. An overview of the EPLD is given in figure 5. Figure 6 shows how the EPLD's are arranged to form two sets of addresses (for double buffering). This architecture allows six pixels to be addressed simultaneously where four of the six pixels are adjacent.

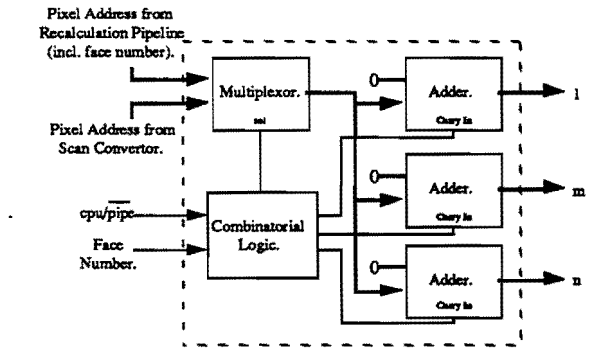

Figure 5. The Memory Addressing EPLD.

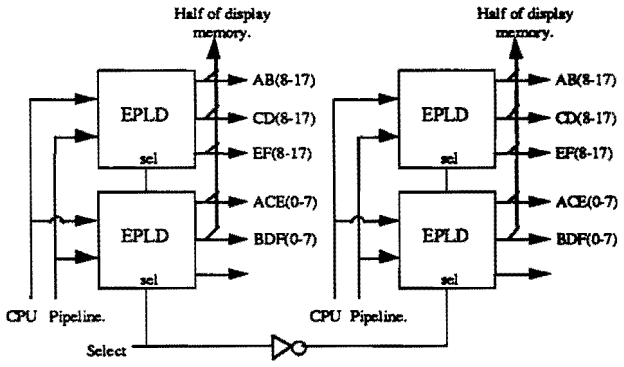

Figure 6. Address formation from the EPLD's.

At the start of each display memory swap the renderer must clear all the RGB and Z components of each pixel in the display memory. As well as being able to read six pixels simultaneously the EPLD may write to six pixels simultaneously. When clearing the display memory contents after a double buffer swap the scan converter may write the same initialisation values to six pixels at one time. As a result the display memory clear time is reduced by a factor of six.

5.0 Conclusion.

We have outlined the principles of operation of an address recalculation pipeline and its benefits for virtual reality applications. The nature of its hardware architecture and its operation requires a complex arrangement of display memories. The needs of the anti-aliasing hardware also complicate the design of the display memories which must be optimised for performance and for efficiency since their cost dominate the system.

In this paper we have described the complex addressing schemes which form a vital contribution to the success of our address recalculation pipeline and make it a very attractive platform on which to base virtual reality systems.

Acknowledgments.

Matthew Regan acknowledges the support of a priority Australian Postgraduate Research Award. This work has been supported by an Australian Research Council small research grant.

References.

[1] Regan, M. and Pose, R. "Priority Rendering with Virtual Reality Address Recalculation Pipeline". Siggraph 94 (to appear).

[2] Regan, M. and Pose, R. "An Interactive Graphics Display Architecture". In proceedings of IEEE Virtual Reality Annual International Symposium, VRAIS 93, (Sept 18-23, 1993) Seattle WA. pp. 293-299

[3] Regan, M. and Pose, R. "A Low Latency Virtual Reality Display System". Monash University, Department of Computer Science, Technical Report TR166, September 1992.

[4] Pose, R. and Regan, M. "Techniques for reducing Virtual Reality Latency with Architectual Support". Proceedings of the East West Conference on Multimedia, Hypermedia and Virtual Reality, (Sept 14-16, 1994), Moscow (to appear).

[5] Robinett, W., and Roland, J., "A Computational Model for the Stereoscopic Optics for Head Mounted Display". In Presence 1, (winter 1992), pp. 45-62.

[6] Deering, M., "High Resolution Virtual Reality", Siggraph 92, pp. 195-202.

[7] Wolberg, G. "Digital Image Warping" IEEE Computer Society Press, 1990.

[8] Cypress Semiconductor CMOS Data Book 1988, CY7C516, pp. 5.71-5.82.