White Paper

Perspectives on Image Quality in the Silicon Graphics® Onyx2® InfiniteReality® Graphics System

Alan Dare
SGI
Mountain View, California
United States
March 16, 2000

Introduction

The following paper describes some of the features of the Silicon Graphics Onyx2 InfiniteReality graphics system. Specifically, how these features affect image quality and processing is discussed.

The Pipeline

The InfiniteReality graphics system shown in Figure 1 is composed of a set of three to six boards attached to an Onyx2 computer system. The application running on the Onyx2 CPU(s) transfers graphics commands directly to the InfiniteReality graphics system, freeing the CPU for application processing. The InfiniteReality graphics

Figure 1

system comprises three or more boards: a Geometry Engine® processor, one or more Raster Managers, and the Display Generator. The Geometry Engine processor provides geometry transformations, lighting calculations, and image processing functions. The Raster Manager performs pixel operations. The Raster Manager also provides color and transparency blending and contains memory for storing textures (images). The Display Manager converts the digital data from the Raster Manager to an analog video signal. The Display Generator is programmable and can support different portions of the digital buffer and a variety of video signals (640x480, 1280x1024, 1920x1200, and so on). This pipeline architecture provides flexibility and high throughput. The InfiniteReality graphics system can generate up to 11 million polygons per second and 896 million pixels per second.

Hardware Resolution

Onyx2 InfiniteReality supports a wide range of display resolutions. The base system includes one Raster Manager board and provides 2.62 million pixels. Two Raster Manager and four Raster Manager systems offer 5.24 million and 10.48 million pixels, respectively. This allows InfiniteReality to support multiple high-resolution monitors. Pixel processors support 10-bit RGB and 12-bit RGBA components for pixels, providing a selection of more than 68 billion colors for any given object. Sixteen-bit luminance alpha is also supported by the video display subsystem. Applications may select luminance to be displayed as color-mapped pseudocolor or as a static-gray X-visual. Ultrahigh color resolution (16 bits) is supported for rendering operations such as texture (images) using luminance. This accurate color or luminance information is critical for image processing and is maintained throughout the graphics subsystem. Other commercial off-the-shelf (COTS) imaging systems do not maintain this resolution in their graphics subsystems. By decreasing the precision of the imagery in the graphics subsystem, detail in the imagery is lost. Performing image processing functions on imagery with truncated resolution can introduce unexpected anomalies, further compounding the problem.

Modern displays can reproduce 256 shades (8 bits of information) of gray. The display's or CRT's photometric response to input stimulus is known as its gamma response characteristic (the red or top line in Figure 2). The ideal CRT would provide a linear photometric response to its input (dashed line in Figure 2). With a nonlinear output, the monitor introduces distortion in the displayed image. An inverse gamma response function (green or bottom line in Figure 2) applied to the input can correct for the nonlinearities of the monitor. This is called gamma correction. Since the input is discontinuous (containing only integer values 1, 3, 4, and so on) a look up table (LUT) can be used to implement the gamma correction (shown in Figure 3). There are as many entries in the LUT as there are possible input values. An input value is used as an index into the LUT and the value at that location is output. The value in the LUT may increase or decrease the input value to compensate for the nonlinearities of the monitor. The InfiniteReality graphics system maintains a full 12 bits for the LUT to reduce distortion.

The last step in displaying images on the screen is converting the digital image into an analog video signal. The InfiniteReality Display Generator uses high-performance digital to analog converter (DACs) with a 220M pixel per second bandwidth. The ability to maintain 12-bit RGBA (color) or 16-bit luminance through the image processing chain makes InfiniteReality the highest resolution integrated COTS graphics solution available.

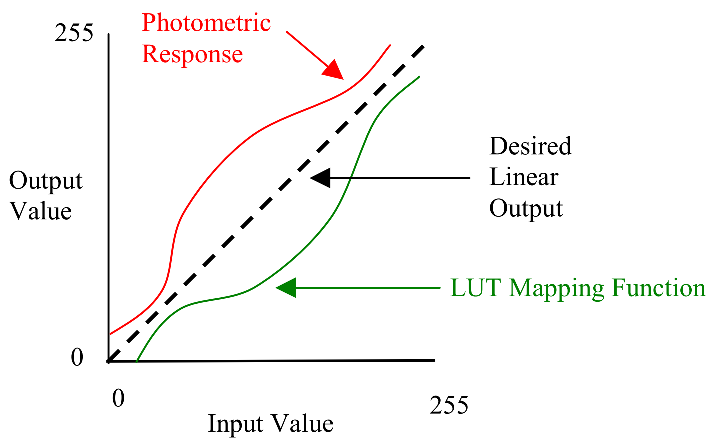
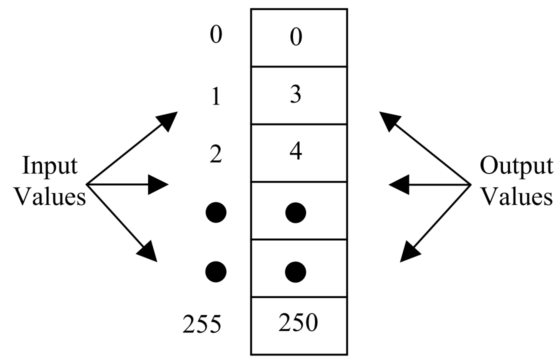


Figure 2



Look Up Table (LUT)
Figure 3

Convolutions

A convolution is a generic image processing operation that involves the multiplication of an image by an array of coefficients or weighting factors referred to as a kernel. By changing the coefficients used in a kernel, an image may be enhanced with low or high pass filtering, may have noises removed, or may have edge detection performed. Figure 4a defines a 3x3 kernel of equal weighting in all locations. Kernels may be of different sizes and are normally composed of odd values (3x3, 5x5, 7x7, and so on). An image is a two-dimensional array of digital numbers (DN), and the image shown in Figure 4b contains nine digital values. Moving the center of the kernel across the image, multiplying each coefficient in the kernel by the corresponding digital number in the original image, and adding all the resulting products performs a convolution. The DN or "pixel" directly under the center of the kernel is replaced with this convoluted value. Figure 5 demonstrates the convolution process. In Figure 5a the kernel is centered over the value in row two, column two. All the digital numbers under the shaded area (kernel) will be multiplied by the corresponding coefficients contained in the kernel. These values are summed and the result replaces the value contained in row two, column two. The kernel is moved to the next digital number in the image and the process is repeated. Figures 5b and 5c demonstrate how the kernel is moved across the two-dimensional array or image to compute convoluted values.

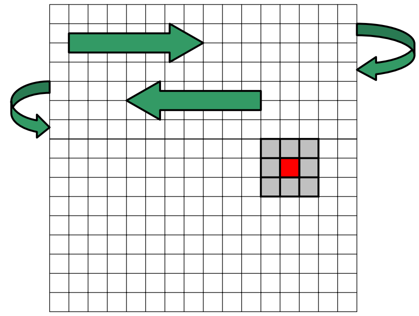
1/9	**1/9**	**1/9**
1/9	**1/9**	**1/9**
1/9	**1/9**	**1/9**

(a) Kernel

72	70	67
67	68	71
71	72	72

(b) Digital Numbers in Original Image

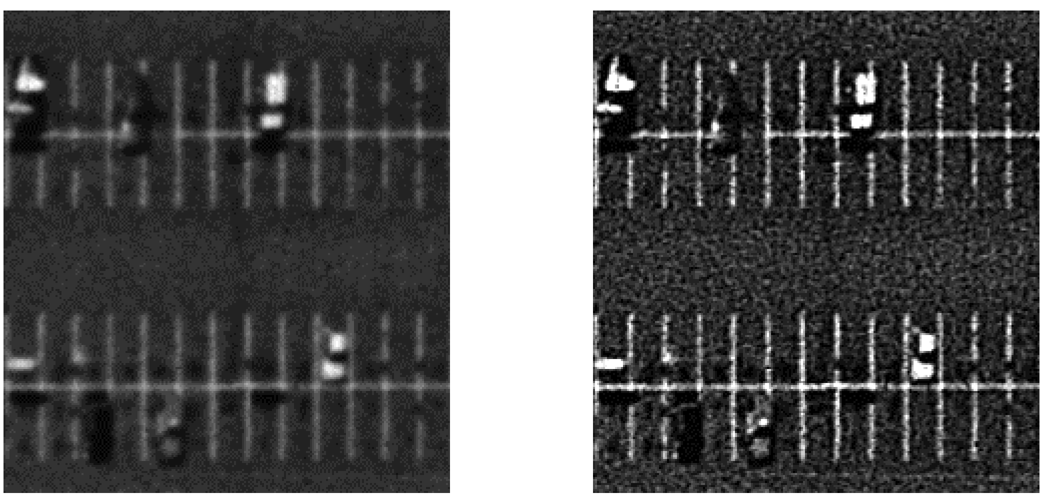
Figure 4



(a) (b) (c)

Figure 5

The effect a convolution has on an image depends on the size of the kernel and the values of the coefficients used in the kernel. When a sharpening enhancement is performed, larger kernels pick up finer detail in the image. Figure 6 illustrates the dramatic difference in sharpening using a 3x3 kernel and a 7x7 kernel on the same image. However, the drawback to using larger kernels in image processing is the increased computational requirements. The computational requirements in using a 7x7 convolution kernel are approximately five times greater than using a 3x3 convolution kernel. Due to the cost of building graphics systems capable of performing a 7x7 convolution in real time, most computer systems employ a 3x3 kernel in hardware. To put this in perspective, an IDEX class image workstation used by the U.S. Department of Defense employs a 5x5 convolution kernel in hardware to achieve superior image analysis.



3x3 Kernel 7x7 Kernel

Figure 6

Resampling

When an image is rotated, enlarged, zoomed, draped over a digital elevation model, or otherwise corrected for geometric distortion, a process of interpolation is required. Figure 7 depicts a common problem in resampling. A rotated or skewed image, shown by the dotted grid, must be converted to an unrotated or unskewed image (shown by the solid line grid). Resampling can be thought of as a convolution of a distorted image with a moving window function. However, in resampling the output is computed *between* the original pixels. Thus, the resampling weighting function is continuous, unlike the discrete array used in a convolution. To demonstrate the issue, look at the green square belonging to a corrected image in Figure 7. What value from the rotated or distorted images belongs in the green square? The green square covers most, but not all, of the rotated image's cell containing the value X. It also covers part of three adjacent pixels whose value is W.

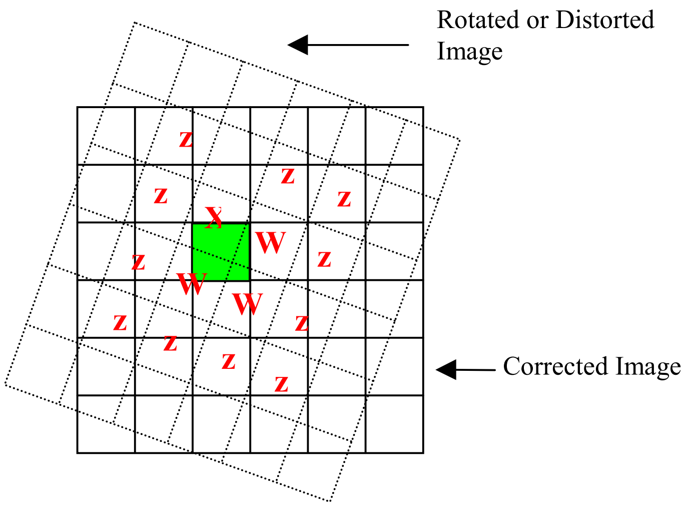
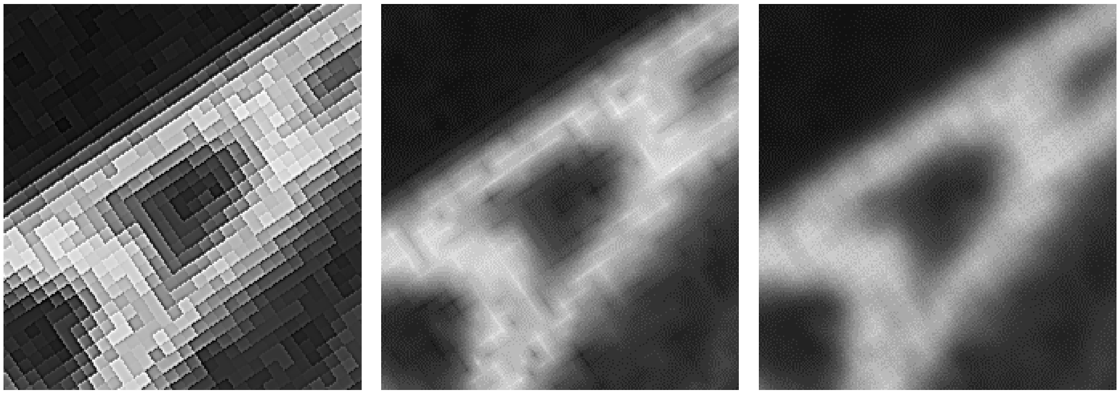


Figure 7

There are three commonly used methods to determine what value belongs in the green square: nearest neighbor, bilinear interpolation, and bicubic interpolation. Nearest neighbor is computationally the simplest method. The value belonging in the green square would be the value of the closest pixel in the input image, disregarding the slight offset. With nearest neighbor, the value in the green square would be X. It is the closest value to the green square. Notice that the cell directly above the green square in the corrected image also covers a large portion of the underlying cell containing the value X. In nearest neighbor resampling, it too would contain an X. Thus, two locations would contain the value X in the corrected image. However, there was only one pixel in the original image that contained the value X. Nearest neighbor resampling produces the poorest quality output. Due to "roundoff," nearest neighbor resampling introduces up to plus or minus one-half a pixel in geometric distortion. Figure 8 contains an image that has been enlarged and rotated. In defense and medical imaging, an estimation of the volume contained within the borders of the white lines may be necessary. In Figure 8, it is difficult to determine the exact borders that contain the volume in the image processed using nearest neighbor interpolation.

The second resampling technique is bilinear interpolation. Bilinear interpolation takes a distance-weighted average of the digital numbers of the four nearest pixels. Thus, using bilinear interpolation the green square in Figure 7 would contain a value that was composed of the cell containing X and the three cells containing W. The exact value is a percentage of each of the four pixels surrounding the green cell. The percentage of each value used is based on the distance from the green cell. Figure 8 illustrates the dramatic difference between nearest neighbor and bilinear interpolation. The image processed with bilinear interpolation is much smoother in appearance than the image processed with nearest neighbor interpolation. The determination of volume with the bilinear interpolated image is clearer. However, it lacks sharpness in its definition.

The bicubic interpolated image shown on the far right in Figure 8 has increased the sharpness and definition of the volume and white lines. Bicubic interpolation takes the distance-weighted average of the 16 nearest pixels. It produces a superior image at the cost of increased processing. This is of benefit in the defense and medical imaging community. As anticipated, bicubic interpolation is far more computationally complex than bilinear interpolation. This computational requirement limits most graphics systems and is not commonly seen in commercial graphics systems. However, bicubic interpolation is built into the hardware of an IDEX class image workstation used by the U.S. Department of Defense and the Onyx2 InfiniteReality system. By including bicubic interpolation in hardware, high image quality can be maintained on rotated and zoomed imagery without sacrificing electronic light table performance.



Nearest Neighbor Bilinear Bicubic

Figure 8

Conclusions

A significant investment was made in the quality and precision handling capability of the Onyx2 InfiniteReality hardware. Onyx2 InfiniteReality is capable of maintaining full 12-bit RGBA or 16-bit luminance images throughout the image processing pipeline. In the image pipeline, the system can perform 7x7 convolutions and bicubic interpolations to preserve the image quality out to the gamma LUTs for the monitor. Combining this quality and precision with the processing power of the graphics pipe makes the Onyx2 InfiniteReality graphics system the highest quality and highest performance COTS graphics system available.

3180[10/01] J13210