HP Color Recovery Technology

HP Color Recovery is a technique that brings true color capability to interactive, entry-level graphics devices having only eight color planes.

by Anthony C. Barkans

For many years the only practical way to display high-quality true color images was on a computer with a graphics subsystem providing at least 24 color planes (see the definition of true color on page 52). However, because of the high cost of color graphics devices with 24 planes, many users chose 8-plane systems. Unfortunately, using these 8-plane systems required giving up some color capabilities to save cost.

HP has developed a technique called HP Color Recovery which provides a method for displaying millions of colors within the cost constraints of an 8-plane system. For an example of the image quality provided by HP Color Recovery consider Fig. 1. Fig. 1a shows a close up of a jet plane stored as a full 24-bit-per-pixel true color image. Fig. 1b shows the same jet plane displayed using a traditional 8-bit-per-pixel

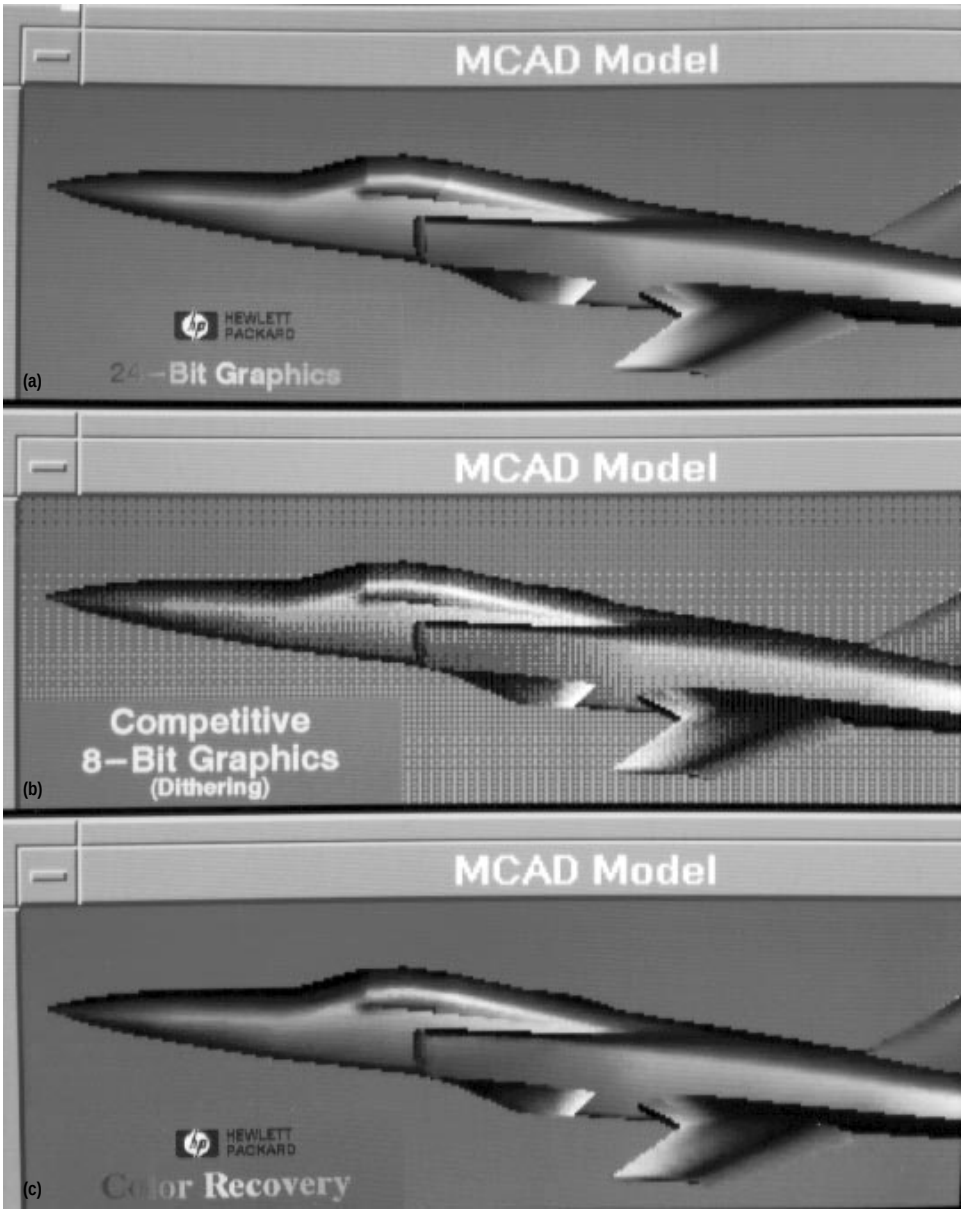

Fig. 1. A true color image and its dithered representations. (a) True color 24-bit image. (b) Typical eight-bit graphics dithered image. (c) An HP Color Recovery dithered image.

True Color

In this paper the term true color is used to define color reproduction such that the underlying digital quantization of the color within an image is not discernable by the human eye. In other words a continuous spectrum of color, such as in a rainbow, can be be displayed so that the color appears to vary continuously across the image. In most computer graphics systems this is accomplished using 24 bits of color information per pixel. With 24 bits, any single pixel can be displayed at one of 2^{24} (16.7 million) colors.

Some graphics systems may define true color to be represented by less than 24 bits per pixel.

system. Finally, Fig. 1c shows how the jet plane will be displayed when using HP Color Recovery in an 8-bit-per-pixel mode on the HCRX-8 graphics device.

Of course, pretty pictures aren't enough. Therefore, one of the primary design goals for HP Color Recovery was to supply the additional color capabilities without giving up interactive performance. Another goal was to be able to work with all types of applications running in a windowed environment such as the X Window System and HP VUE. The implementation of HP Color Recovery used in current HP workstations meets these goals.

Traditional Eight-Plane Systems

Traditional eight-plane systems can display only 256 colors. Two approaches have been employed to get the best results with limited colors. The first is called either *pseudo color* or *indexed color*. This method selects a set of 256 colors and then limits the application to using only that fixed set of colors. For many applications, such as word processing and business graphics, this approach works reasonably well. This is because the resultant images are made up of very few colors. However, when an application needs more than 256 colors, such as realistically shaded MCAD (mechanical computer-aided design) images or human faces in video sequences, then another approach is needed. Since more than 256 colors are required for these applications, a technique to simulate more colors is used. For these applications a technique called *dithering* is employed. The idea of dither is to approximate a single color by displaying two other colors at intermixed pixel locations. For example, a grid of black and white pixels can be displayed to simulate gray. Such a grid of black and white pixels will indeed look gray when viewed from a distance. The primary problem with dithering is that since most people tend to work close to the display, dithered images are viewed as having a grainy or textured appearance (see Fig. 1b).

Color Theory and Dither

Before discussing the details of how HP Color Recovery works, an overview of color theory as it relates to computer generated images and dither should be helpful. This overview describes how the human eye is tricked into seeing color, color precision in graphics, and a dithering method.

Tricking the Human Eye

It is often noted that computer monitors use red, green, and blue (RGB) to produce true color images. A reasonable question to ask is: "Why use these particular colors?" If one examines the spectrum of visible light, it can be seen that red is at the end of the spectrum with the longest wavelengths that the human eye can see while blue is at the other end. Note that green is in about the middle. Also note that white is a mix of all colors. Therefore by mixing varying amounts of red, green, and blue any color can be created. For example, forcing both the red and the green CRT beams to be on at any single location will result in a dot that appears yellow to the human eye.

Thus, one can create the visual appearance of any color by mixing the red, green, and blue components at any pixel location. However, it is interesting to note that the human eye can also perceive a new color when the component colors are mixed spatially. For example, a checkerboard of red and green pixels will be perceived as yellow when viewed from a distance. It is this spatial mixing of color to form a new color that is exploited by dither.

Color Precision

In most systems that deal with true color, color is specified to eight bits for each of the three color components: red, green, and blue. The choice of eight bits is based on two factors. First, the human eye cannot distinguish an infinite number of shades because the dynamic range of the eye is limited. For the most part shaded surfaces rendered with eight bits per color appear smooth with the underlying quantization not readily apparent to the viewer. The second factor that works in favor of using eight bits per color component as a standard is that eight-bit bytes are very convenient to work with in a computer system.

Simple Dithering

When using a 24-bit color system, any displayable color component can be specified using eight bits. For example, consider the red component. When there is no red in a pixel the red component is specified with a binary value of 00000000, which is a decimal 0. A full bright red is specified as a binary number 11111111, which is decimal 255. Of course, high-end display systems, such as the HCRX-48Z, use 24 bits to store and display true color information. The visual quality of these high-end displays is shown in Fig. 1a. However, since low-cost systems typically have a total of only eight bits per pixel to store the color information, an approximation to the true color image is made. The most common method is dither using three bits each for the red and green components. This leaves two bits for blue. Using fewer bits for blue is based on the fact that the human eye has less sensitivity to blue. With fewer bits available per color component, the quantization of the colors becomes apparent to the viewer. The effects of using a limited number of bits for each color can be seen in Fig. 1b.

Dithering approximates any color by using a combination of colors at adjacent pixels. When viewed from a distance the

image appears to be the correct color. However, since dithered systems can store only a limited number of bits in the frame buffer, the primary task of the dithering logic is to select the best set of values to use.

For dithering purposes it is convenient to think of each eight-bit binary component of color as a number in a *three point five* (3.5) representation. This representation means there are three bits on the left side of the binary point and five bits on the right side of the binary point. For example, assume the true color value for red is given as the binary number 01011000. In a 3.5 representation the number becomes 010.11000 binary, which is 2.75 decimal. Since the final dithered values can only be three-bit integers, it can be seen that using only numbers two and three would be desirable. Ideally, the dither would set 3/4 of the pixels to three and 1/4 to two.

If we consider the original color component as being an eight-bit value in a 3.5 format, then the dither values stored in the dither table should be evenly spaced between 000.00000 and 000.11111 (decimal 0.0 to almost decimal 1.0). The output of the table is added to the original eight-bit color component. Once the addition is complete the value is truncated to the desired number of bits for storage in the frame buffer. As a simple example assume that we are continuing to work with a red component that is originally specified as the binary number 01011000. In addition assume that we are using a 2 × 2 dither to reduce the original 8-bit color component to three bits. (The notation 2 × 2 dither means that the dither pattern will repeat in a 2 × 2 grid across the image.) To use a 2 × 2 dither, the least-significant bits of the X and Y window addresses of the pixel are used to index the dither table. The following example shows how a 2 × 2 dither is applied to one pixel of the true color value for red. Table I represents the values in a 2 × 2 dither table.

Table I
A 2 × 2 Dither Table

Indexes		Dither Value	
LSB of Y	LSB of X	Binary	Decimal
0	0	.10100	.625
0	1	.01100	.375
1	0	.00100	.125
1	1	.11100	.875

At the upper left of the window the X and Y addresses are both 0. To dither the data for this pixel location using our color value for red we do the following:

	Binary	Decimal
Input color	010.11000	2.750
Dither value (from Table I)	+ .10100	+ .625
Result	011.01100	3.375
Truncated three-bit value	011	3

Therefore at address 0,0 we would store a 011 binary in the frame buffer for red. Applying the above dither would result in three of the four pixels within every four-pixel block being stored in the frame buffer with a value of 011 (see Fig. 2). The fourth pixel in each block, the one with the LSB of Y set to a 1 and the LSB of X set to a 0, will have a 010 stored in the frame buffer. When a region of this color of red is viewed from a distance the color would appear to be the correct value of 010.11000. If the dithered jet plane shown in Fig. 1b is examined, it can be seen that it is dithered using a method similar to the one described above.

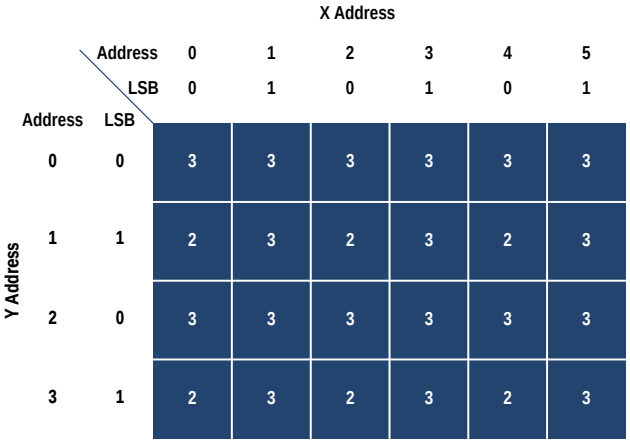

Fig. 2. Results after applying dither. Each box represents a pixel location on the display screen. For example, address (0,0) is defined as the upper left corner of the display. Also note that the numbers stored at each pixel location represent the results of applying the dither values given in Table I to a red component of color originally specified as 01011000 binary (2.75 in our 3.5 notation).

From a distance the colors in the dithered image are integrated by the eye so that they appear correct. However, the fundamental problem with dither is that most dithered images are viewed up close and so the dithering pattern is noticeable in the image.

Dithering Is Key

It is important to realize that to approximate any true color value, a spatial region of the screen is required. This often leads people to say that dithering is a method that trades off spatial resolution for color resolution. However, this is misleading. Some people believe that a single-pixel object cannot be dithered. Actually a single-pixel object can be dithered. The result is that the object will be one of the two dither colors. Going back to the example above, a single-pixel red object specified as binary 01011000 (decimal 2.75) will be stored at any single pixel location as either binary 010 or 011 (decimal two or three). Taken by itself, any single pixel is not a perfect approximation of the true color. However, it is still a reasonable approximation.

The idea of being able to encode each pixel in the image independently by using dither is key to enabling color recovery to work in an interactive environment. As a historical note it should be mentioned that over the last few years several people have developed methods to bring true color capabilities to eight-bit graphics devices. However, these attempts have been based on complex multipixel encoding

schemes. For the most part they have applied data compression techniques to the data stored in the frame buffer. These methods have produced high-quality images, but the encoding is so complex that the user must give up interactive performance to use them. Because of the performance problems these methods have not been widely adopted by the computer graphics community.

HP Color Recovery

The simplest explanation of HP Color Recovery is that it performs the task your eye is asked to do with an ordinary dithered system. In essence, an HP Color Recovery system takes 24-bit true color data generated by an application and dithers it down to eight bits for storage in the frame buffer. Then as the frame buffer data is scanned from the frame buffer to the display, it passes through specialized digital signal processing (DSP) hardware where the work of producing millions of colors is performed. The output of the DSP hardware is sent to the display where millions of colors can be viewed. It is important to recognize that since the data stored in the HP Color Recovery frame buffer is dithered, thousands of applications can work with it. It is also important to recognize that these applications will run at full performance in an interactive windowed environment. In other words, applications do not need to be changed to take advantage of HP Color Recovery.

The Process

HP Color Recovery is a two-part process. First, true color information generated by the application is dithered and then stored in the frame buffer. The type of application generating the true color information is immaterial. For example, true color data can be generated by a CAD application program or as part of a video sequence. The dithering may be done in a software device driver or in the hardware of a graphics controller. It is very important to note that each pixel is treated independently. This pixel independence is key to the ability to work within an interactive windowed environment. The second part of the HP Color Recovery process is to filter the dithered data. The filter is placed between the output of the frame buffer and the DACs that drive the monitor. Fig. 3 shows the HP Color Recovery process starting from when an application generates true color data to when the image appears on the screen. Note that "application" refers to any program that generates true color data for display.

After the application generates the data, it is sent to the device driver. The function of the driver is to isolate the application from hardware dependencies. The driver is supplied by HP. It causes hardware dithering to be used when possible. However, there are times when the driver must perform the dither in software. It is important to note that compared to other dithered systems, there is no performance penalty suffered by an application using HP Color Recovery dither.

The frame buffer stores the image data. Note that in most current systems the output of the dithered frame buffer is sent to the display, resulting in the common patterned appearance in the image. However, with HP Color Recovery, as the frame buffer data is scanned, it is sent through a specialized digital signal processing (DSP) circuit. The DSP is a sophisticated circuit that removes the patterning from the dithered image stored in the frame buffer. This circuit performs over nine billion operations per second. Despite this enormous amount of processing the circuit is surprisingly small. It is this small size that makes HP Color Recovery inexpensive enough to be considered for inclusion in low-end graphics systems.

The Dither Process

In HP Color Recovery the quality of the displayed image depends on the dither used to encode the image. During the development of HP Color Recovery it was found that the size of the dither region determines how well a color can be recovered. It was found that from a region of 2^N pixels the technique can recover about N bits of color per component. Therefore, an eight-bit frame buffer that stores data in 3-3-2 format (3 bits each for red and green and 2 bits for blue) would need a dither region of 32 pixels for each color component to recover 5 additional bits. Thus, using a 32-pixel dither region, an area in the image of uniform color can have the same visual quality as an 8-8-7 image. For example, the sky behind the jet plane in Fig. 1c was recovered to within 1 bit of the original 24-bit true color data shown in the top image.

Most dithers use a 4×4 dither region. Since a 4×4 region covers only 16 pixels, a larger dither region is needed for HP Color Recovery. Therefore, a dither table with 32 entries organized as 2×16 was selected. (The reason for this odd shape is discussed later in this paper.) In addition, most dithers are as simple as the one described earlier in this paper. However, there are cases in which a simple dither does not

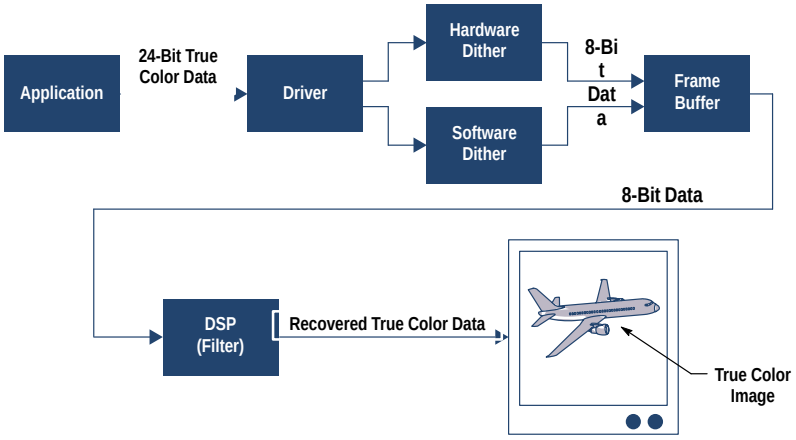

Fig. 3. HP Color Recovery process flow.

work well. Note that in using the simple dither method described above, all true color values from binary 11100000 to 11111111 would dither to 111. For HP Color Recovery the dither table includes both positive and negative numbers. This improves the color range over which the dither is useful.

The HP Color Recovery dither is a little different from most dithers. However, it is on the same order of complexity. It should also be noted that the HP Color Recovery dither is included in the hardware of all HP graphics workstations that support this technique. This means that using the HP Color Recovery dither does not cause a decrease in performance.

The Filter Process

In the example given earlier a red color component represented by the binary value 01011000 (2.75 in decimal) was used to illustrate simple dithering. For this example we used a 2 × 2 dither region in which the end result of the dither was that 3/4 of the pixels stored in the frame buffer were set to 3 (011) and 1/4 of the pixels were set to 2 (010). It is easy to see that if we average the four pixels in the 2 × 2 region we will recover the original color. This can be done as follows:

([value_1 × number_set_to_value_1] + [value_2 × number_set_to_value_2])/total_number_pixels

Using the example data we obtain: ([3 × 3] + [2 × 1])/4 = 2.75.

This averaging works very well in regions of constant color, such as the sky behind the jet plane in Fig. 1. However, there is one fundamental issue that must be addressed for HP Color Recovery to be viable and that is how to handle edges in the image. If edges are not accounted for then the resultant image will blur. The two-dimensional representations of an area of a display screen shown in Fig. 4 are used to illustrate the problem of edge detection and the way the problem is addressed in HP Color Recovery.

As in Fig. 2, each box represents a pixel location on the display screen. In Fig. 4a the numbers represent the original true color data for one of the color components (e.g., red) in a 24-bit per pixel system. Fig. 4b shows the same region after simple dithering has been applied. Fig. 4c shows the pixel values after the application of HP Color Recovery. Fig. 4c pixel values represent the color data that would be displayed on the computer screen.

Region A in each of these figures is an area of constant color, whereas region B encompasses an edge. For illustration purposes, the dither region is again assumed to be 2 × 2 pixels.

The dithered color data shown in Fig. 4b is derived from the original color data shown in Fig. 4a and from using the simple dithering technique described in connection with Table I. The data shown in Fig. 4b is what would be stored in the frame buffer and displayed in a typical dithered system (e.g., Fig. 1b).

When it is time to display Pix_1 the data for the four pixels shown as Region A in Fig. 4b would be sent to the filter. The data stored in the region would be summed and then divided by the number of pixels in the region. The sum of the pixels in Region A is 11 and 11/4 = 2.75. Thus, the output of the filter when evaluating Pix_1 would be 2.75. This output value would be displayed on the computer display at Pix_1's location. Note that the output of the filter is the exact value of the original data at that point in Fig. 4a.

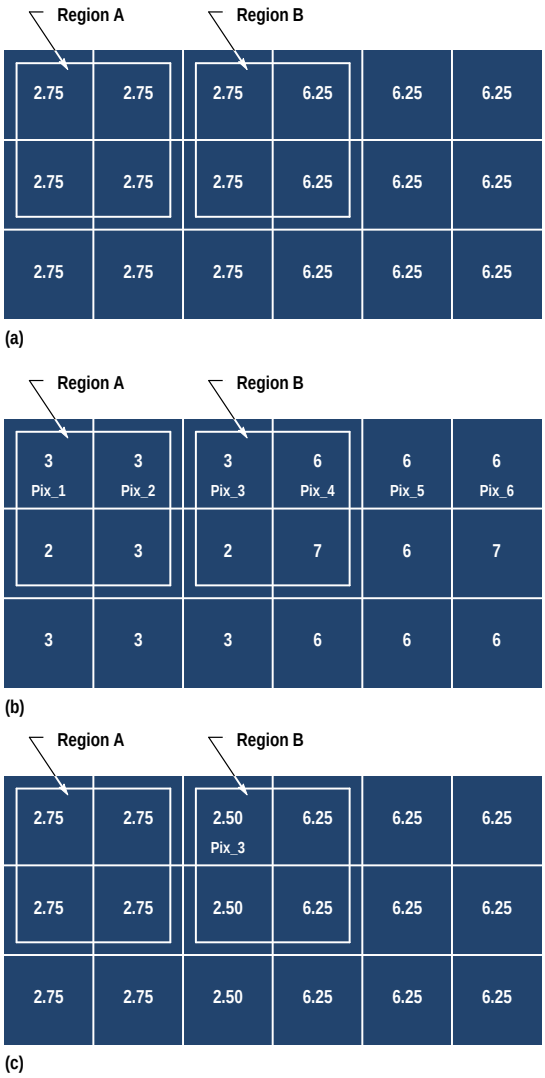

Fig. 4. (a) Pixel values for the original 24–bit per pixel color data. (b) The color data from Fig. 4a after it has been dithered and placed in the frame buffer. This would be the data displayed in a typical dithered system with the result appearing as in Fig. 1b. (c) The pixels from Fig. 4b after applying HP Color Recovery.

The next pixel along the scan line to be evaluated is Pix_2. The filter region for evaluating Pix_2 would include the two rightmost pixels of region A and the two leftmost pixels of region B (see Fig. 4b). Applying the filter operation for Pix_2 again results in the output value matching the value at that location in Fig. 4a (2.75).

If the evaluation is done on Pix_3, the pixels in region B would be summed and then divided by the number of pixels in the region, and the result would be 4.50. This value is very different from the original data value of 2.75 in Fig. 4a. Using the value of 4.50 at Pix_3 would result in edge smearing. To solve this problem a special edge detector that looks for edges in noisy data is used. The idea is to compare each pixel in the filter region with a value that is within ±1 of the pixel being evaluated. Since the data stored at Pix_3 is a 3,

only pixels within region B that have a value of 2, 3, or 4 would pass the edge compare. The values that pass the edge detector are then summed and the total is divided by the number of pixels that pass the edge compare. For Pix_3, only Pix_3 and the pixel below it would pass the edge detector. Summing the two passing values together and dividing by 2 gives a result of 2.50. This value is slightly different from the original value of 2.75, but it is a better estimation to the original than the 4.50 obtained without the edge detection. The displayed values for the entire example region are shown in Fig. 4c.

Software Considerations

Since dithered frame buffers are in common use today, many existing software applications can work with a dithered frame buffer. All of these applications could work with HP Color Recovery.

On products that use the Model 712's graphics chip, which is described in the article on page 43, and HP's Hyperdrive (HCRX), HP Color Recovery is supported. In these products we have chosen to have HP Color Recovery enabled as the default for 3D applications run in an eight-bit visual environment. Thus when using the 3D graphics libraries Starbase, PHIGS, or PEXlib, and opening an application in an eight-bit visual environment with true color mode, HP Color Recovery will normally be enabled. Of course, setting an application to use a pseudo color map will disable HP Color Recovery and give the application the desired pseudo color capability. Because Xlib is tied into the pseudo color model rather than the 3D libraries, Xlib applications leave HP Color Recovery off by default. However, a mechanism is supported that allows HP Color Recovery to be enabled when using Xlib.[1] The biggest change is that Xlib applications must do their own dithering.

Implementation

The implementation of HP Color Recovery was based on the assumption that color recovery would be most useful in entry-level graphics products. Entry-level graphics products are defined as products in which there is storage for only 8 bits per pixel in the frame buffer. These same products that benefit the most from HP Color Recovery are also the ones where product cost must be carefully controlled. Therefore, the implementation effort was driven with a strong sense of cost versus end user benefit.

Dither Table Shape. As mentioned earlier, the dither region shape used with HP Color Recovery is 2×16. The optimum shape would be closer to square, such as 4×8. However, the filter circuit needs storage for the pixels within the region. A 2×16 circuit requires that the current scan line's pixel and the data for the scan line above be available. This means that as data for any scan line enters the circuit, it is used to evaluate pixels on the current scan line. In addition, the data is saved in a scan line buffer so it can be used when evaluating the pixels on the next scan line (see Fig. 5). It should be noted that the storage for a scan line of data uses approximately one half of the circuit area in the current implementation. Therefore, if a 4×8 region had been used, three scan

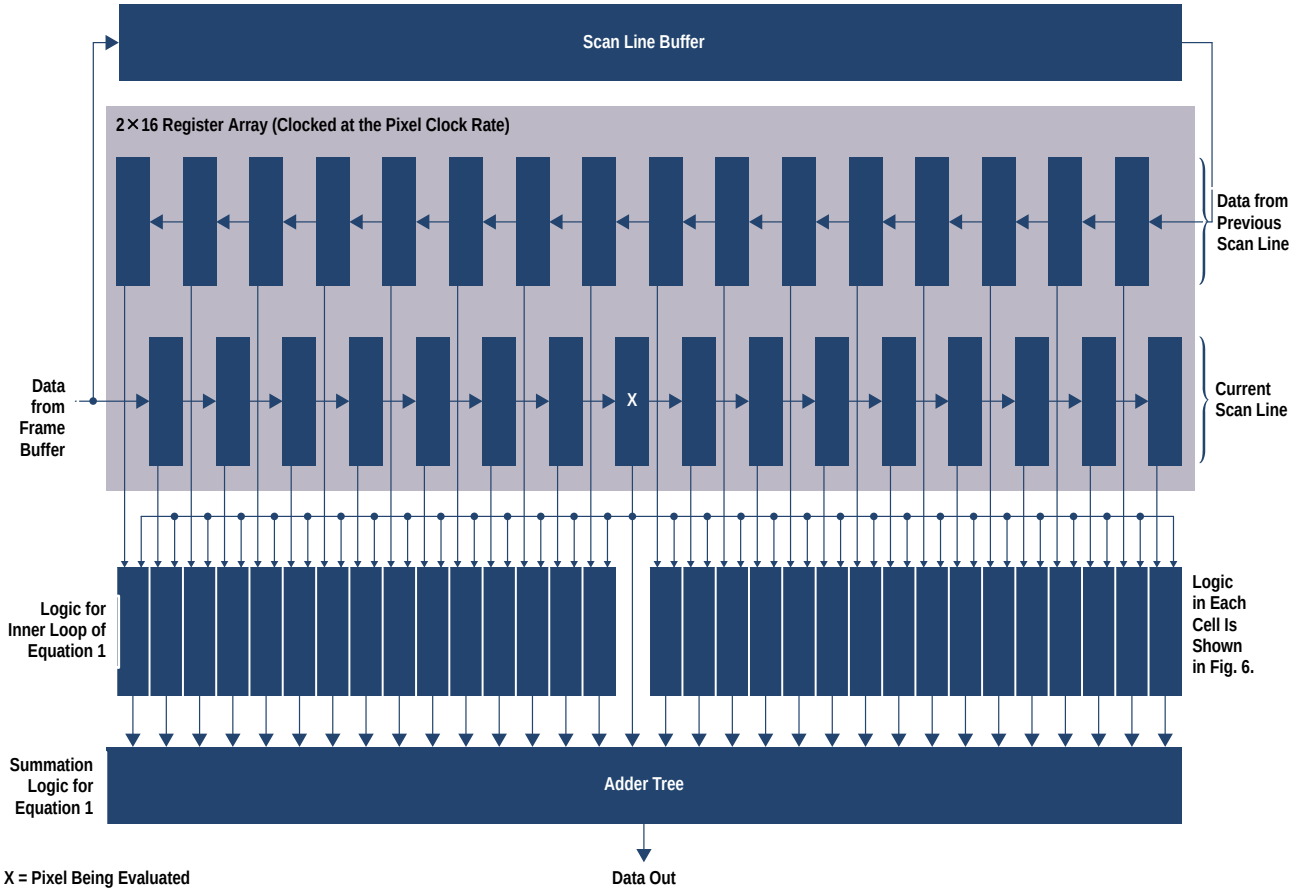

Fig. 5. A block diagram of the HP Color Recovery filter circuit.

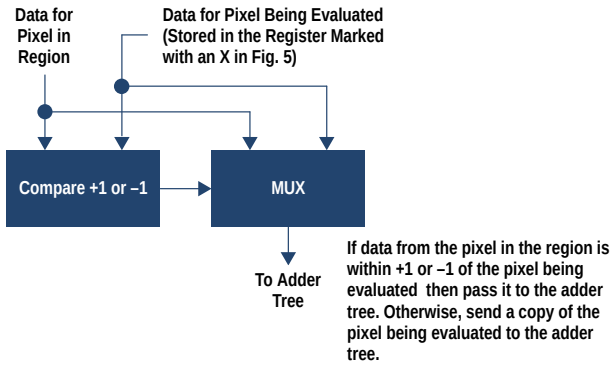

Fig. 6. A simplified representation of the logic circuitry that exists in each of the logic blocks in Fig. 5.

line buffers would have been required, almost doubling the cost of the HP Color Recovery logic.

Filter Function Logic. As explained earlier, the HP Color Recovery filter function averages the data within a region by summing the data for the pixels that pass an edge compare operation. The sum is then divided by the number of pixels that pass the edge compare. Typically, building the logic for a filter function like this is difficult and costly because it requires a divide circuit running at the video clock rate of 135 MHz. The HP Color Recovery filter function is implemented so that this is not a problem.

The implementation details of the filter function are complex. However, if we ignore the high-speed pipeline issues and some minor adjustments required to optimize image quality, we can reduce the implementation of the filter function to the following equation:

$$\sum_{i=0}^{k-1}(\text{Frame\_Buffer\_Data}_i)(W_i) + (\text{Evaluated\_Pixel})(\overline{W}_1) \quad (1)$$

where k is the number of pixels in the filter and W_i is a flag equal to one when a pixel passes an edge compare operation and zero when it doesn't. This flag can be thought of as the output of the comparator shown in Fig. 6.

The idea behind this equation is that if a pixel passes the edge compare, include it in the total. On the other hand, if a pixel fails the edge compare, then substitute the data for the pixel being evaluated for the failing pixel. The overriding assumption is that the pixel being evaluated is a reasonably good guess of the true color data. The worst case is that all the pixels around the sample fail the edge compare and the dithered color is used for that location. Since dithering uses a reasonable sample at each location this extreme case results in a reasonable image being displayed.

To see how this works let's look at two examples. In the first example assume that the pixel being evaluated is a single red dot specified using 01011000 binary (2.75 in our decimal numbering system). This is the same color used in some of the examples described earlier. However, this time let us assume that it is dithered to a value of 011. Also assume that this pixel is surrounded by green. Since the edge compare is done on a per-color basis, all the pixels in the region except the pixel being evaluated will fail the edge compare. In this case we will add a red value of 011 thirty-two times. The result out of the adder tree in Fig. 5 will have a red value of 01100000 (3.00 in decimal). Although this is not exact it will appear as a red dot in the middle of a green region. In other words, a reasonable approximation.

In the second example assume a region that is filled with red is specified with the same eight-bit binary value of 01011000. Also assume the simple dither method described earlier is used. In this case 3/4 of the pixels will be stored as 011. The other 1/4 will be stored as 010. Since none of the pixels fails the edge compare we will send twenty-four pixels with the value of 011 and eight with the value 010 to the adder tree. The results of the adder will be a binary value of 01011000 (2.75 decimal). In this case the output of HP Color Recovery will match the input true color data exactly.

Hardware details. The filtering logic, which was shown in a systems context in Fig. 3, is expanded in Fig. 5. As the frame buffer is scanned, each pixel in the display is sequentially sent to the logic shown in Fig. 5. The left side of the figure shows the path taken as the data for each pixel read from the frame buffer enters the filtering logic. The data is sent both to a pipeline register for immediate use, and to a scan line buffer for use when the next scan line is being evaluated. The 32 registers shown in Fig. 5 store the data for the 2 × 16 region being evaluated. These registers are clocked at the pixel clock rate. Note that the data for each pixel on the display will pass through the location marked with the X. When a pixel is at the location X, it is called the pixel being evaluated. This means that the results of applying equation 1 are assigned to the display at the screen address of X.

The 32 pixels stored in the pipeline registers shown in Fig. 5 are sent through blocks of logic that perform the inner loop evaluation of equation 1. This inner loop is essentially an edge detector. The logic shown in Fig. 6 allows only pixels that have similar numeric values to the pixel being evaluated to be included in the summation. The summation logic is simply an adder tree that sums the results of the pixels passing the edge compare. The filter function is performed in parallel for all the pixels within the filter region.

Given the complexity of the function being performed in the filter circuit, the circuit is surprisingly small. The entire filter circuit is made up of approximately 35,000 transistors. Compared to the number of transistors required to increase the number of color planes, this is very small. For example increasing the number of color planes from 8 to 16 on a typical SVGA (Super VGA) system (1024 × 768-pixel resolution) requires over 8,000,000 transistors, which is 1M bytes of additional frame buffer memory. Because of the small size of the HP Color Recovery circuit, it is inexpensive enough to be included in entry-level graphics systems.

Questions and Answers

Thus far the concepts behind HP Color Recovery have been discussed. It has been shown that HP Color Recovery can supply additional color capabilities to low-end graphics systems while maintaining an interactive windowed environment. The following are answers to the most frequently asked questions about the practical use of HP Color Recovery.

- Question: Is there a difference between a 24-bit true color image and one displayed using HP Color Recovery?

Answer: Yes. If you view a 24-bit image and an HP Color Recovery image side by side there are differences. For example, the back edge of the wing in Fig. 1c has some artifacts in it. At normal size the artifacts can be found but are less noticeable than in Fig. 1c.

- Question: How many colors are reproducible with HP Color Recovery?

  Answer: In the best case HP Color Recovery can provide up to 23 bits of accuracy. However, in typical images about four million colors can be reproduced.

- Question: Are artifacts introduced by HP Color Recovery?

  Answer: In areas of very low contrast, artifacts will show up. Again the back edge of the wing in Fig. 1c is a good example.

- Question: Does HP Color Recovery look the same on all HP products that support it?

  Answer: No. The first implementation was designed for the graphics chip used in the HP 9000 Model 712 workstation. After that design was finished some improvements were made which ended up in the HCRX family of graphics devices. These changes are hidden deep in the details of the implementation, enabling any application using HP Color Recovery on one product to work without change on the other products.

- Question: Do applications need to change to use HP Color Recovery?

  Answer: If the application was written using a 3D application program interface the answer is no. Of course it must be running in an eight-bit visual environment on a device that supports HP Color Recovery. In addition, the application must have been written to use the 24-bit true color model. However, if the application was written using Xlib then it must be changed to do the dithering. Details can be found in reference 1.

- Question: Is there a way to turn HP Color Recovery off?

  Answer: Yes. Set the environment variable `HP_DISABLE_COLOR_RECOVERY` to any value.

- Question: What happens to the color map in the HP 9000 Model 712's graphics chip when HP Color Recovery is enabled?

  Answer: In the graphics chip there are two hardware color maps. By default, the X11 server permanently downloads the default color map into one of these hardware color maps. If HP Color Recovery is enabled the remaining color map is used by HP Color Recovery. See the article on page 43 for more information about these color maps.

- Question: What happens to the color map on HCRX graphics when HP Color Recovery is enabled?

  Answer: On HCRX graphics devices there are two hardware color maps in the overlay planes and two in the image planes. By default, the X11 server permanently downloads the default color map into one of the overlay planes' hardware color maps. This is true in each of the following configurations:
  - The HCRX-8 and HCRX-8Z frame buffer configurations with no transparency have one hardware color map in the overlay planes and two in the image planes that are available. In this configuration the HP Color Recovery color map can be downloaded into any of the available hardware color maps.
  - The HCRX-8 and HCRX-8Z frame buffer configurations with transparency have only one hardware color map in the overlay planes and only one in the image planes. Since the hardware color map for the overlay planes already has the default color map loaded into it, there is only one color map available for HP Color Recovery to choose from. Therefore, in this configuration the HP Color Recovery color map is downloaded into the remaining hardware color map.
  - The HCRX-24 and HCRX-24Z frame buffer configurations with or without transparency have one hardware color map in the overlay planes and two in the image planes that are available. In this configuration, when using an eight-bit visual depth the HP Color Recovery color map can be downloaded into any of the available hardware color maps.

- Question: Does HP Color Recovery work with logical raster operations?

  Answer: Yes. Like any dithered frame buffer system, HP Color Recovery works with raster operations such as `AND`, `OR`, and `XOR`.

- Question: How do image processing applications interact with HP Color Recovery?

  Answer: There are two basic classes of image processing applications: feature finding and image enhancement.
  - Feature finding. Most feature-finding applications are based on edge detection. The results of running one of these types of applications can be displayed using HP Color Recovery. However, as with other dithered frame buffers, any application using the frame buffer as the image source may have problems if it does not account for the dither.
  - Image enhancement. Image enhancement applications are typically used to enhance images for the human visual system. The goal of many of these applications is to bring out low-level features of the image. It is possible to preprocess the image and send it to HP Color Recovery. However, if there is a need for an extremely high-quality image (e.g., medical imaging) a 24-bit frame buffer may be necessary.

- Question: If an image is dithered using a dither method other than the one developed for HP Color Recovery, can it be displayed on a system that supports HP Color Recovery?

  Answer: Yes. One option is to turn HP Color Recovery off. However, the image can be processed with HP Color Recovery on. In this case the image will be viewable. The image quality will be comparable to viewing the image on a typical dithered system, but the dithering artifacts will be replaced with a new set of artifacts.

- Question: Can an image created using the HP Color Recovery dither method be viewed on an eight-bit system that does not support HP Color Recovery?

  Answer: Yes. However, it is important to realize that without the HP Color Recovery back end the dithering artifacts will be visible in the image.

- Question: Can a user read the frame buffer data?

Answer: Yes. However, as with any dithered system there is the issue of precision. For example, if the red data is generated with eight bits of precision, then the readback will give a three-bit dithered value for the data. The data on readback is not the same as the eight-bit value generated by the application.

- Question: Does HP Color Recovery work with multimedia applications?

  Answer: Yes. By removing the dithering artifacts, image quality during MPEG (video) playback is improved.

- Question: Does HP Color Recovery impact application performance?

  Answer: No. The HP Color Recovery dither is implemented in fast hardware in both the Model 712's graphics chip and the HCRX graphics subsystem. When hardware dithering cannot be used, such as with virtual memory double buffering, a software dither is performed by the device driver. Since the dither is the same complexity as common dithers, there is no performance penalty for using HP Color Recovery when compared to using other dithered systems.

  In addition, the DSP circuit in the back end is placed in the path of the data being scanned into the monitor. As such the DSP does get in the path (without affecting application performance) when the system is performing what the user sees as interactive tasks.

- Question: Can an image generated using HP Color Recovery be displayed on output devices other than monitors (e.g., printers)?

  Answer: Many applications generate a print file. In this case the data displayed on the monitor is not used to create the print file. Therefore, HP Color Recovery will not interfere with the output. Another method used to generate hardcopy is a screen dump. Unfortunately, a complete solution for dumping a color-recovered image to a printer is not available yet.

Conclusion

Color recovery brings added color capabilities to entry-level systems. Since the technology is based on dither, these additional color capabilities can be brought to an entry-level system while maintaining an interactive environment that supports many current applications.

Acknowledgments

Many people have helped transform HP Color Recovery from an idea into a reality. The list would be too long to print here. Without the list of names I hope everyone involved knows that I appreciate their efforts. However, there are several people that I must list by name. These people are Paul Martin, Larry Thayer, Brian Miller, and Randy Fiscus. Additionally, a special thanks goes to Dave McAllister who took my notes and turned them into real logic. Along the way, Dave found many innovations that led to a better design.

Reference

1. *HP Color Recovery Technology*, HP publication Number 5962-9835E.