T. Porter & T. Duff - Compositing Digital Images (Page 2)

Next Prev

and Weimar's system reduces all objects to horizontal spans which are composited using a Warnock-like algorithm. In Crow's system a supervisory process decides the order in which to combine images created by independent special-purpose rendering processes. The imaging system of Warnock and Wyatt [5] incorporates 1-bit mattes. The Hanna-Barbera cartoon animation system [4] incorporates soft-edge mattes, representing the opacity information in a less convenient manner than that proposed here. The present paper presents guidelines for rendering elements and introduces the algebra for compositing.

2. The Alpha Channel

A separate component is needed to retain the matte information, the extent of coverage of an element at a pixel. In a full color rendering of an element, the RGB components retain only the color. In order to place the element over an arbitrary background, a mixing factor is required at every pixel to control the linear interpolation of foreground and background colors. In general, there is no way to encode this component as part of the color information. For anti-aliasing purposes, this mixing factor needs to be of comparable resolution to the color channels. Let us call this an *alpha* channel, and let us treat an alpha of 0 to indicate no coverage, 1 to mean full coverage, with fractions corresponding to partial coverage.

In an environment where the compositing of elements is required, we see the need for an alpha channel as an integral part of all pictures. Because mattes are naturally computed along with the picture, a separate alpha component in the frame buffer is appropriate. Off-line storage of alpha information along with color works conveniently into run-length encoding schemes because the alpha information tends to abide by the same runs.

What is the meaning of the quadruple (r,g,b,α) at a pixel? How do we express that a pixel is half covered by a full red object? One obvious suggestion is to assign (1,0,0,.5) to that pixel: the .5 indicates the coverage and the (1,0,0) is the color. There are a few reasons to dismiss this proposal, the most severe being that all compositing operations will involve multiplying the 1 in the red channel by the .5 in the alpha channel to compute the red contribution of this object at this pixel. The desire to avoid this multiplication points up a better solution, storing the *pre-multiplied* value in the color component, so that darkened colors for fractional alphas along edges, and black where alpha is 0. Silhouette edges of RGBA elements thus exhibit their anti-aliased nature when viewed on an RGB monitor.

It is important to distinguish between two key pixel representations:

$$black = (0,0,0,1);$$
$$clear = (0,0,0,0).$$

The former pixel is an opaque black; the latter pixel is transparent.

3. RGBA Pictures

If we survey the variety of elements which contribute to a complex animation, we find many complete background images which have an alpha of 1 everywhere. Among foreground elements, we find that the color components roll off in step with the alpha channel, leaving large areas of transparency. Mattes, colorless stencils used for controlling the compositing of other elements, have 0 in their RGB components. Off-line storage of RGBA pictures should therefore provide the natural data compression for handling the RGB pixels of backgrounds, RGBA pixels of foregrounds, and A pixels of mattes.

There are some objections to computing with these RGBA pictures. Storage of the color components premultiplied by the alpha would seem to unduly quantize the color resolution, especially as alpha approaches 0. However, because any compositing of the picture will require that multiplication anyway, storage of the product forces only a very minor loss of precision in this regard. Color extraction, to compute in a different color space for example, becomes more difficult. We must recover $(r/\alpha,\ g/\alpha,\ b/\alpha)$, and once again, as alpha approaches 0, the precision falls off sharply. For our applications, this has yet to affect us.

4. The Algebra of Compositing

Given this standard of RGBA pictures, let us examine how compositing works. We shall do this by enumerating the complete set of binary compositing operations. For each of these, we shall present a formula for computing the contribution of each of two input pictures to the output composite at each pixel. We shall pay particular attention to the output pixels, to see that they remain pre-multiplied by their alpha.

(.5,0,0,.5) will indicate a full red object half covering a pixel.

The quadruple (r,g,b,α) indicates that the pixel is α covered by the color (r/α, g/α, b/α). A quadruple where the alpha component is less than a color component indicates a color outside the [0,1] interval, which is somewhat unusual. We will see later that luminescent objects can be usefully represented in this way. For the representation of normal objects, an alpha of 0 at a pixel generally forces the color components to be 0. Thus the RGB channels record the true colors where alpha is 1, linearly

4.1. Assumptions

When blending pictures together, we do not have information about overlap of coverage information within a pixel; all we have is an alpha value. When we consider the mixing of two pictures at a pixel, we must make some assumption about the interplay of the two alpha values. In order to examine that interplay, let us first consider the overlap of two semi-transparent elements like haze, then consider the overlap of two opaque, hard-edged elements.