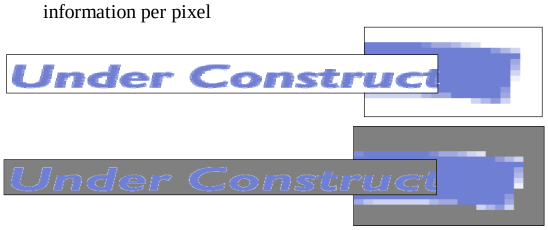
Image Compositing

Compositing Motivation

- Sometimes, a single image needs to be constructed out of parts.
  - Mixing 3D graphics with film
  - adding a backdrop to a scene
  - Painting objects into a scene
- Sometimes, it's just better to do things in parts
  - Can save time in rendering
  - A small problem in one part can easily be fixed in the final image
- Need a method for building up an image from a set of components
  - Ideally, invent a general "algebra" of **compositing**

Image Matting

- To assemble images from parts, we associate a **matte** with each part
  - Record which pixels belong to the foreground, which to the background
  - Discard background pixels when assembling
- Problem: The matte must record more than a single bit of information per pixel



The Alpha Channel

- To make compositing work, we store an alpha value along with color information for every pixel.
- α records how much a pixel is covered by the given color
  - The set of alpha values for an image is called the **alpha channel**
  - Transparent when $\alpha = 0$
  - Opaque when $\alpha = 1$
- Relationship between α and RGB:
  - computed at same time
  - Need comparable resolution
  - Can manipulate in almost exactly the same way

The Meaning of Alpha

- How might we store the information for a pixel that's 50% covered by red?

$$[1 \quad 0 \quad 0 \quad .5]$$

- It turns out that we'll always want to multiply the color components by α, so store (R,G,B,α) in premultiplied form:

$$\begin{bmatrix} \frac{R}{\alpha} & \frac{G}{\alpha} & \frac{B}{\alpha} & \alpha \end{bmatrix}$$

- What do the premultiplied R, G and B values look like?
- What does (0,0,0,1) represent?
- What about (0,0,0,0)?

Compositing Assumptions

- The goal of compositing is to approximate the behaviour of overlaid images inside partially-covered pixels
  - We don't know *how* the pixel is covered, just how much
  - We need to make assumptions about the nature of this coverage
- We'll consider two cases:
  - Two semi-transparent objects; alpha channel records transparency
  - Two hard-edged opaque objects; alpha channel records coverage

Compositing Semi-Transparent Objects

- If we wish to composite two semi-transparent pixels over a background, things are a little easier.
- Suppose we wish to composite colors A and B with opacities α_A and α_B over a background G
- How much of G shows through A and B?

$$(1-\alpha_A)(1-\alpha_B)$$

- How much of G is blocked by A and passed by B?

$$\alpha_A(1-\alpha_B)$$

- How much of G is blocked by B and passed by A?

$$(1-\alpha_A)\alpha_B$$

- How much of G is blocked by A and B?

$$\alpha_A\alpha_B$$

Compositing Opaque Objects

- Assume that a pixel is partially covered by two objects, A and B.
  - We can use α_A and α_B to encode what fractions of the pixel are covered by A and B respectively
- How does A divide the pixel?

$$\alpha_A : (1-\alpha_A)$$

- How does B divide the pixel?

$$\alpha_B : (1-\alpha_B)$$

- *How does A divide B?*

$$\alpha_A : (1-\alpha_A)?$$

- Compositing assumption: A and B are uncorrelated
  - This lets us make educated guesses about the color of the composed pixel
  - Works well in practice

Pixel Pieces

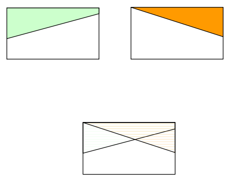
- Given the compositing assumption, we can state the areas of different parts of the pixel:

$\bar{A} \cap \bar{B}$

$\bar{A} \cap B$

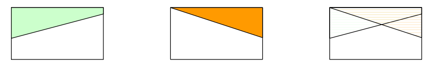
$A \cap \bar{B}$

$A \cap B$



- Why do these areas depend on lack of correlation?

Compositing Possibilities



- The contributions of A and B to the pixel divide the pixel area into four regions. When compositing, we have to choose what will be visible in each region.

Name	*Description*	*Possibilities*
0	$\bar{A} \cap \bar{B}$	0
A	$A \cap \bar{B}$	$0, A$
B	$\bar{A} \cap B$	$0, B$
AB	$A \cap B$	$0, A, B$

- According to this enumeration, how many binary compositing operators are there?

The 12 Compositing Operators

- We can define a compositing operator by giving a 4-tuple listing what to keep in the regions 0, A, B and AB.

(0,0,0,0)
(0,A,0,A)
(0,0,B,B)
(0,A,B,A)
(0,A,B,B)
(0,0,0,A)
(0,0,0,B)
(0,A,0,0)
(0,0,B,0)
(0,0,B,A)
(0,A,0,B)
(0,A,B,0)

Computing the color

- Let's say we want to show a fraction F_A of A and a fraction F_B of B in the composite.
- What should the alpha value of the composite be?

$$\alpha_O = F_A \alpha_A + F_B \alpha_B$$

- What should the color component be in each channel?

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix}_O = F_A \begin{bmatrix} R \\ G \\ B \end{bmatrix}_A + F_B \begin{bmatrix} R \\ G \\ B \end{bmatrix}_B$$

The “plus” operator

- All the operators are all-or-nothing in region AB. Sometimes we want to show a blend of A and B in AB, for example when dissolving from one image to another.
- We define A **plus** B using the tuple (0,A,B,AB) where AB represents a blend of A and B.

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix}_O = \begin{bmatrix} R \\ G \\ B \end{bmatrix}_A + \begin{bmatrix} R \\ G \\ B \end{bmatrix}_B$$

Computing F_A and F_B

- All that remains is to compute F_A and F_B.
  - Depends on and determines the compositing operator
  - Can be derived by inspection of the compositing diagrams

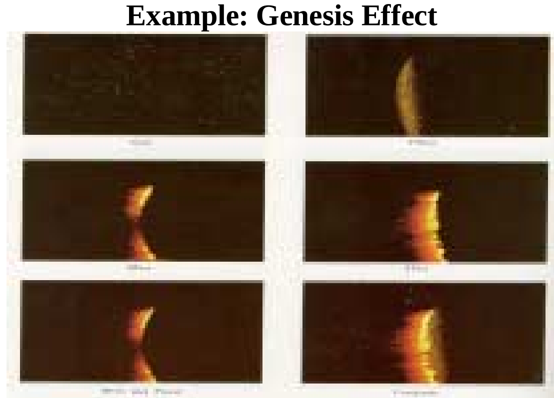
Operation	F_A	F_B
clear		
A		
B		
A over B		
A in B		
A plus B		

Unary Operators

- There are also some useful unary operators

$$darken([R,G,B,\alpha],\phi) = [\phi R, \phi G, \phi B, \alpha]$$
$$dissolve([R,G,B,\alpha],\delta) = [\delta R, \delta G, \delta B, \delta\alpha]$$

Example: Genesis Effect



(*FFire* **plus** (*BFire* **out** *Planet*)) **over darken**(*Planet, 0.8*) **over** *Stars*

Summary

- Reasons for doing compositing
- The meaning of alpha and the alpha channel
- Definition of compositing operators
- Definition and implications of the compositing assumption
- Computation of composited images
- Practical use of compositing