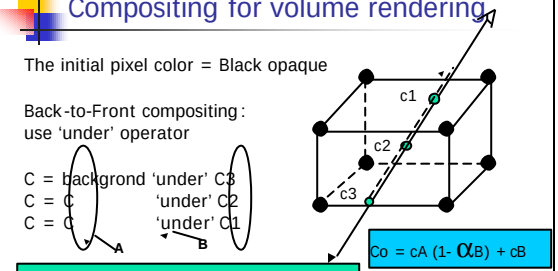
# Digital Image Compositing

## Compositing for volume rendering

The initial pixel color = Black opaque

Back-to-Front compositing:
use 'under' operator

C = background 'under' C3
C = C 'under' C2
C = C 'under' C1




Co = cA (1- αB) + cB

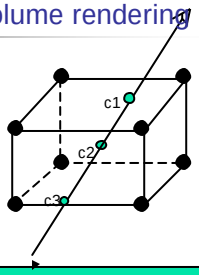
Cout = Cin * (1-α(x)) + C(x)* α(x)

## Compositing for volume rendering

Or you can use 'Front-to-Back' Compositing formula

Front-to-Back compositing:
use 'over' operator

C = background 'over' C1
C = C 'over' C2
C = C 'over' C3

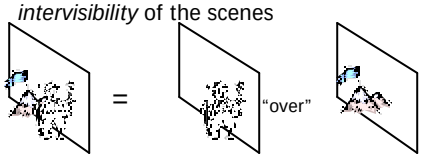



...

Cout = Cin + C(x)* α(x)*(1 - αin); αout = αin + α(x) *(1 αin)

## What is compositing anyway?

- A method for combining two or more images in a way that approximates the *intervisibility* of the scenes




2 ½ D rendering : scenes have to be disjoint in depth

## Why compositing?

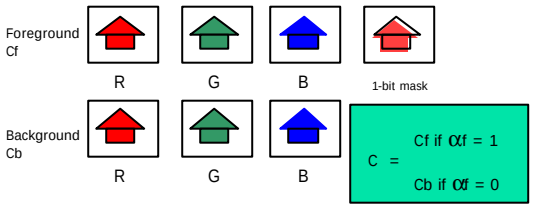
- Special effects (shake hands with important people...)
- Share the load of rendering
- Render translucent objects (translucent polygons, volume rendering, etc)

## How to composite?

- A separate component other than RGB is needed to represent the coverage of en element at a pixel
- This component is called *alpha* channel
  - alpha = 0 -> zero coverage
  - alpha = 1 -> full coverage

## How to composite? (2)

- 1 bit matte




## Alpha Channel

- The value of alpha can be in [0,1] to indicate the extent of the coverage
  (or how opaque the object is)
- A pixel's 'color' is represented by a quadruple $(r,g,b,\alpha)$

  (0,0,0,1) = opaque black
  (0,0,0,0) = transparent

## Alpha Channel (2)

- How to represent a pixel that is half covered by a full red object?
  -> (1,0,0,0.5) ?
  the red contribution is - 1* 0.5
- If we want to composite a foregrond color $C_f$ (1,0,0) over a background color $C_b$ then we do C = (1,0,0) * 0.5 + (1-0.5)*$C_b$
  i.e. $C = C_f * \alpha + (1-\alpha) * C_b$

## Pre-multiplied alpha

Given $C = \underline{C_f * \alpha} + (1-\alpha) * C_b$
Every time we want to perform composite, we need to multiply the color by its alpha
-> why not just pre-multiplied the color components by alpha and stored that way?
$(R,G,B,\alpha)$ –> $(R\alpha, G\alpha, B\alpha, \alpha)$

This way, we have $C = C_f + (1-\alpha)\, C_b$
$(r,g,b,\alpha)$ premultiplied quadruple -> $(r/\alpha, g/\alpha, b/\alpha, \alpha)$
real color

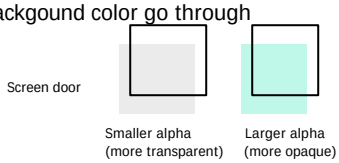
## Compositing Algebra

- Foreground over background is only one of the compositing (the simplest) methods.
- What are the formula for all possible kind of compositing (A over B, A under B, A in B...)?
- The issues is to understand and formulate the interplay between the alpha values of two input picture

## Compositing Algebra (2)

What is alpha any way?

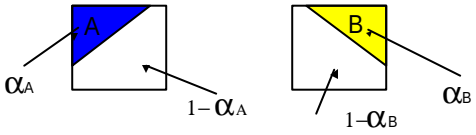
1. Represents the opaqueness of semitransparent objects. With alpha = $\alpha$, the object will let $(1-\alpha)$ of backgound color go through



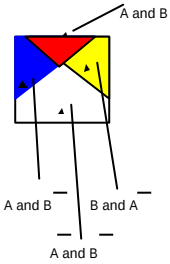
## Compositing Algebra (3)

2. Represents the amount of pixel area covered by the object. (1-a) of the pixel is not covered, and a of the pixel is covered.

(this method is better for understanding this paper)

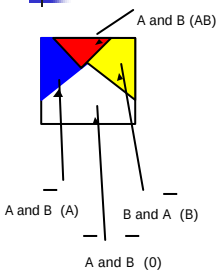



## Compositing Algrbra (4)

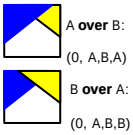



Assumption: If B has alpha value $\alpha_B$, then the area A is Also divided as $\alpha_B$ and $1-\alpha_B$ And vice versa

## Possible Compositing of A,B




All the possible compositing of A and B can be enumerated based on the value in the four regions ( 0, A, B, AB)



A **over** B:
(0, A,B,A)

B **over** A:
(0, A,B,B)

## Compositing Arithmetic

Basic Idea:
To composite A an B: Each input picture source (A or B) will survive in its own matte ($\alpha$), and the fraction (F A) of its own matte not covered in the output picture


Example: A survives in $\alpha_A$, and $(1-\alpha_B)$

So final color $C_o$ = contrib. A + contrib. B

$$Co = \alpha_o \frac{\alpha_A F_A C_A + \alpha_B F_B C_B}{\alpha_A F_A + \alpha_B F_B}$$

## Compositing Arithmetic (2)

$$Co = \alpha_o \frac{\alpha_A F_A C_A + \alpha_B F_B C_B}{\alpha_A F_A + \alpha_B F_B}$$

since $\alpha_o = \alpha_A F_A + \alpha_B F_B$

$cA = CA\,\alpha_A$ and $cB = CB\,\alpha_A$ (cA, cB are alpha premultiplied color)

Co = cA FA + cB FB (note that Co is also alpha premultiplied color)

## Compositing Arithmetic (3)

Co = cA FA + cB FB (note that Co is also alpha premultiplied color)

Example 1: Now let's look at 'Over'

We know FA = 1, FB = $1-\alpha_A$

So we have Co = cA + cB $(1-\alpha_A)$

Example 2: 'Under'

FA = $1-\alpha_B$ FB = 1

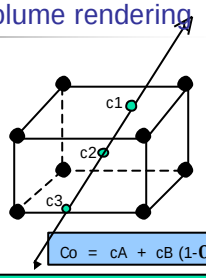
So Co = cA $(1-\alpha_B)$ + cB

## Compositing for volume rendering

Or you can use 'Front-to-Back' Compositing formula

Front-to-Back compositing: use 'over' operator

C = clear 'over' C1
C = C 'over' C2
C = C 'over' C3




Co = cA + cB (1-αA)

...

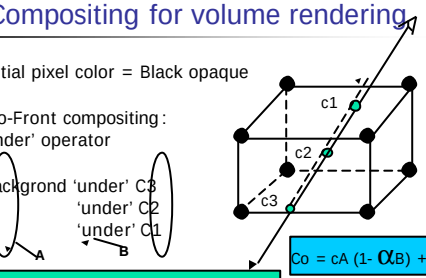
Cout = Cin + C(x)* α(x)*(1 - αin); αout = αin + α(x) *(1-αin)

## Compositing for volume rendering

The initial pixel color = Black opaque

Back-to-Front compositing: use 'under' operator

C = background 'under' C3
C = C 'under' C2
C = C 'under' C1




Co = cA (1- αB) + cB

Cout = Cin * (1-α(x)) + C(x)* α(x)