

Compositing

CS4620 Lecture 23



CORNELL
Habitat for
Humanity

Volunteer to build
roof trusses for
local homes

APRIL 30 - MAY 4, 2018
HO PLAZA



No experience
necessary! Stop by &
join us for as long as
you like!

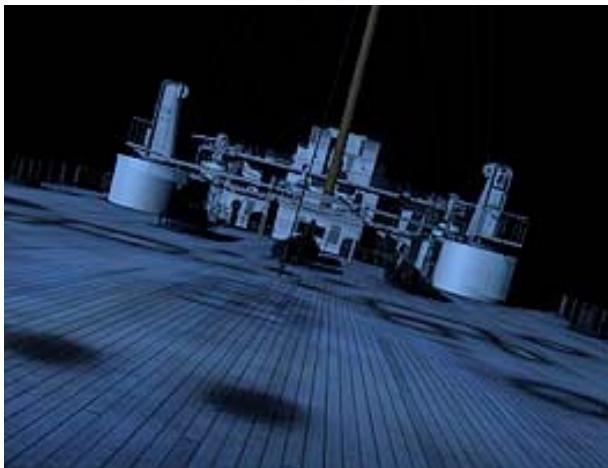
SIGN UP AT
HabitatCornell.org

CONTACT US AT
HabitatForHumanityCornell@gmail.com

Pixel coverage

- Antialiasing and compositing both deal with questions of pixels that contain unresolved detail
- Antialiasing: how to carefully throw away the detail
- Compositing: how to account for the detail when combining images

Compositing



[Titanic ; DigitalDomain; vfxhq.com]

Combining images

- Often useful combine elements of several images
- Trivial example: video crossfade
 - smooth transition from one scene to another



- note: weights sum to 1.0
 - no unexpected brightening or darkening
 - no out-of-range results
- this is *linear interpolation*

$$\begin{aligned}r_C &= tr_A + (1 - t)r_B \\g_C &= tg_A + (1 - t)g_B \\b_C &= tb_A + (1 - t)b_B\end{aligned}$$

Combining images

- Often useful combine elements of several images
- Trivial example: video crossfade
 - smooth transition from one scene to another

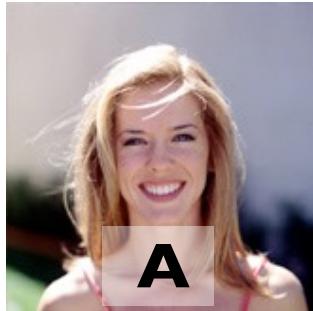


- note: weights sum to 1.0
 - no unexpected brightening or darkening
 - no out-of-range results
- this is *linear interpolation*

$$\begin{aligned}r_C &= tr_A + (1 - t)r_B \\g_C &= tg_A + (1 - t)g_B \\b_C &= tb_A + (1 - t)b_B\end{aligned}$$

Combining images

- Often useful combine elements of several images
- Trivial example: video crossfade
 - smooth transition from one scene to another



- note: weights sum to 1.0
 - no unexpected brightening or darkening
 - no out-of-range results
- this is *linear interpolation*

$$r_C = tr_A + (1 - t)r_B$$

$$g_C = tg_A + (1 - t)g_B$$

$$b_C = tb_A + (1 - t)b_B$$

Combining images

- Often useful combine elements of several images
- Trivial example: video crossfade
 - smooth transition from one scene to another

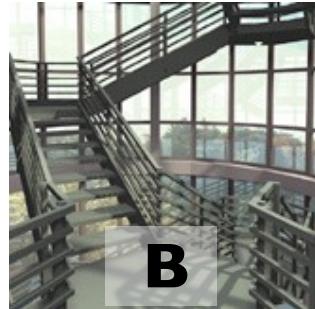
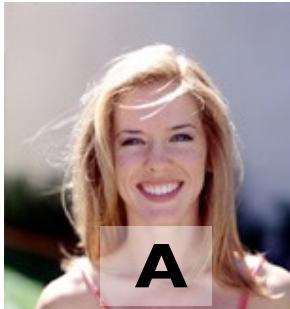


$$r_C = tr_A + (1 - t)r_B$$
$$g_C = tg_A + (1 - t)g_B$$
$$b_C = tb_A + (1 - t)b_B$$

- note: weights sum to 1.0
 - no unexpected brightening or darkening
 - no out-of-range results
- this is *linear interpolation*

Combining images

- Often useful combine elements of several images
- Trivial example: video crossfade
 - smooth transition from one scene to another



- note: weights sum to 1.0
 - no unexpected brightening or darkening
 - no out-of-range results
- this is *linear interpolation*

$$\begin{aligned}r_C &= tr_A + (1 - t)r_B \\g_C &= tg_A + (1 - t)g_B \\b_C &= tb_A + (1 - t)b_B\end{aligned}$$

Foreground and background

- In many cases just adding is not enough
- Example: compositing in film production
 - shoot foreground and background separately
 - also include CG elements
 - this kind of thing has been done in analog for decades
 - how should we do it digitally?

Foreground and background

- How we compute new image varies with position

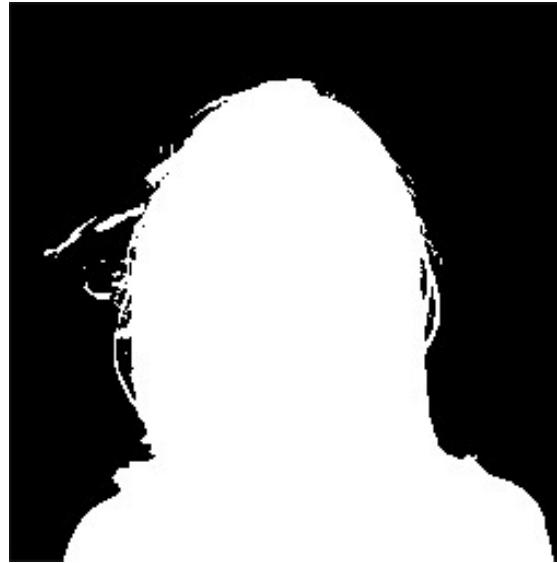
[Chuang et al./Corel]



- Therefore, need to store some kind of tag to say what parts of the image are of interest

Binary image mask

- First idea: store one bit per pixel
 - answers question “is this pixel part of the foreground?”

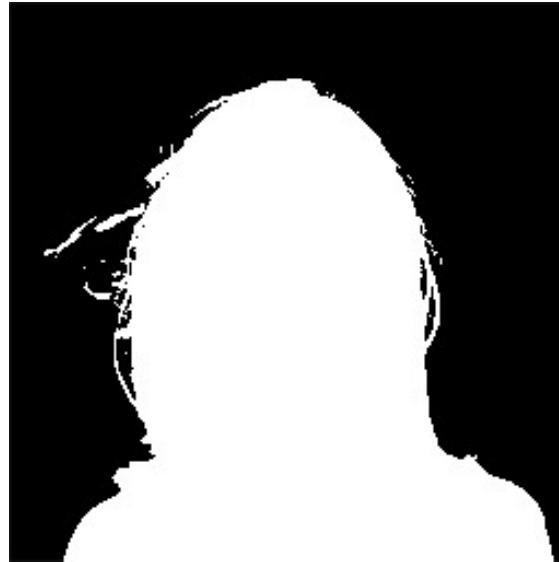


- causes jaggies similar to point-sampled rasterization
- same problem, same solution: intermediate values

Binary image mask

- First idea: store one bit per pixel
 - answers question “is this pixel part of the foreground?”

[Chuang et al. / Corel]

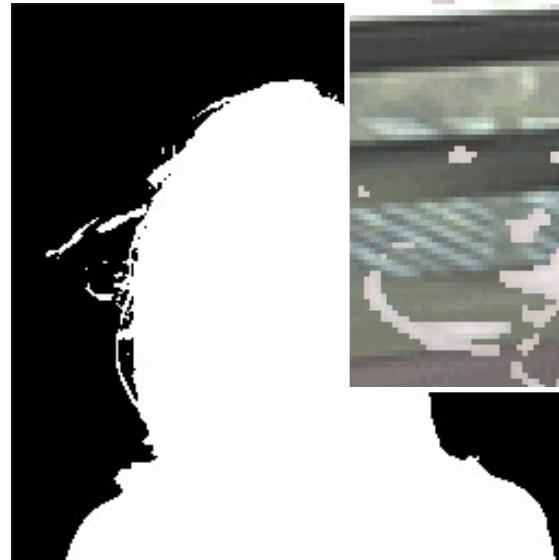


- causes jaggies similar to point-sampled rasterization
- same problem, same solution: intermediate values

Binary image mask

- First idea: store one bit per pixel
 - answers question “is this pixel part of the foreground?”

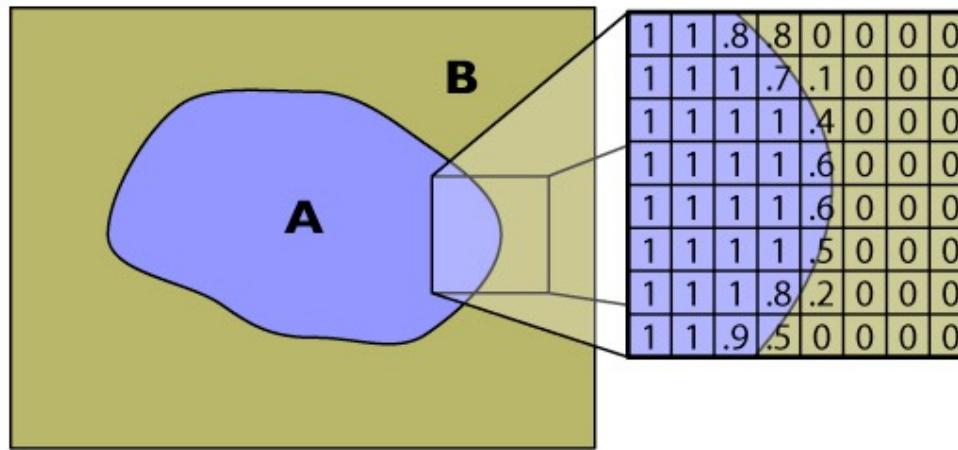
[Chuang et al. / Corel]



- causes jaggies similar to point-sampled rasterization
- same problem, same solution: intermediate values

Partial pixel coverage

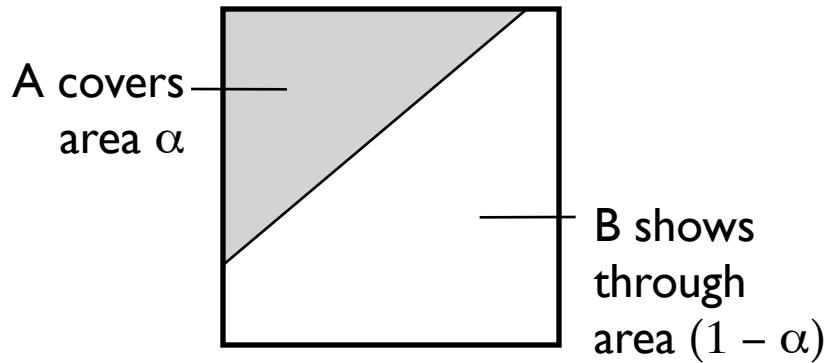
- The problem: pixels near boundary are not strictly foreground or background



- how to represent this simply?
- interpolate boundary pixels between the fg. and bg. colors

Alpha compositing

- Formalized in 1984 by Porter & Duff
- Store fraction of pixel covered, called α



$$E = A \text{ over } B$$

$$r_E = \alpha_A r_A + (1 - \alpha_A) r_B$$

$$g_E = \alpha_A g_A + (1 - \alpha_A) g_B$$

$$b_E = \alpha_A b_A + (1 - \alpha_A) b_B$$

- this exactly like a spatially varying crossfade
- Convenient implementation
 - 8 more bits makes 32
 - 2 multiplies + 1 add per pixel for compositing

A little notation

- Define $\bar{\alpha} = 1 - \alpha$
- then in $E = A$ over B

$$c_E = \alpha_A c_A + \bar{\alpha}_A c_B$$

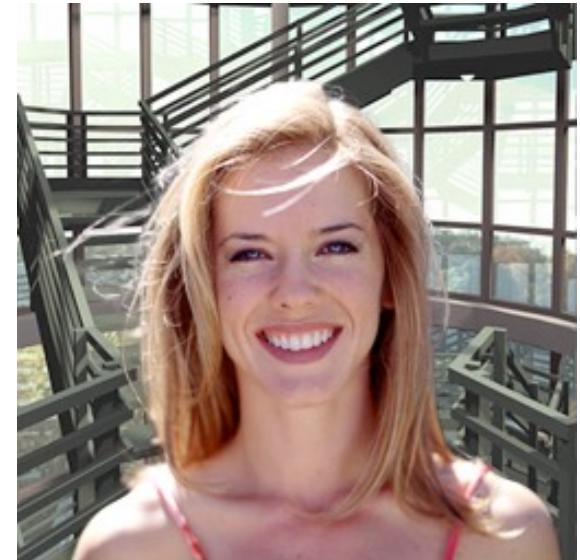
Alpha compositing—example

[Chuang et al. / Corel]



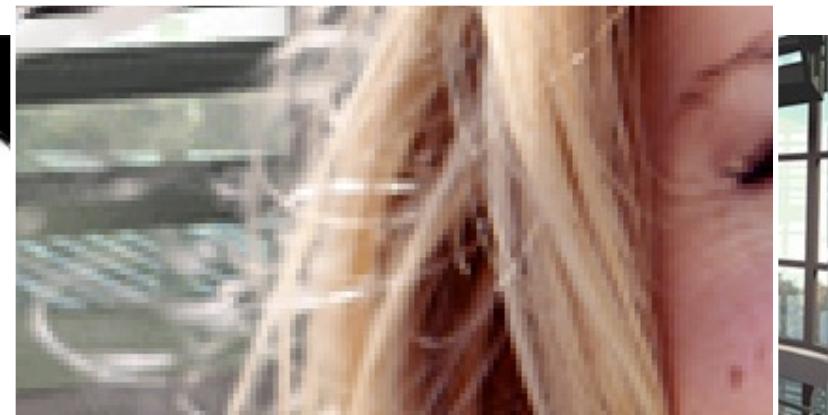
Alpha compositing—example

[Chuang et al. / Corel]



Alpha compositing—example

[Chuang et al. / Corel]

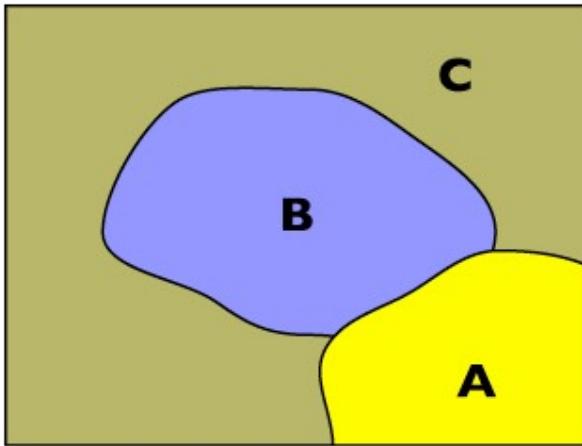


Compositing composites

- so far have only considered single fg. over single bg.
- in real applications we have n layers
 - *Titanic* example
 - compositing foregrounds to create new foregrounds
 - what to do with α ?
 - want (c_A, α_A) over $(c_B, \alpha_B) = (\alpha_A c_A + \bar{\alpha}_A c_B, ???)$
- desirable property: associativity
 - A over $(B$ over $C) = (A$ over $B)$ over C
 - to make this work we need to be careful about how α is computed

Compositing composites

- Some pixels are partly covered in more than one layer

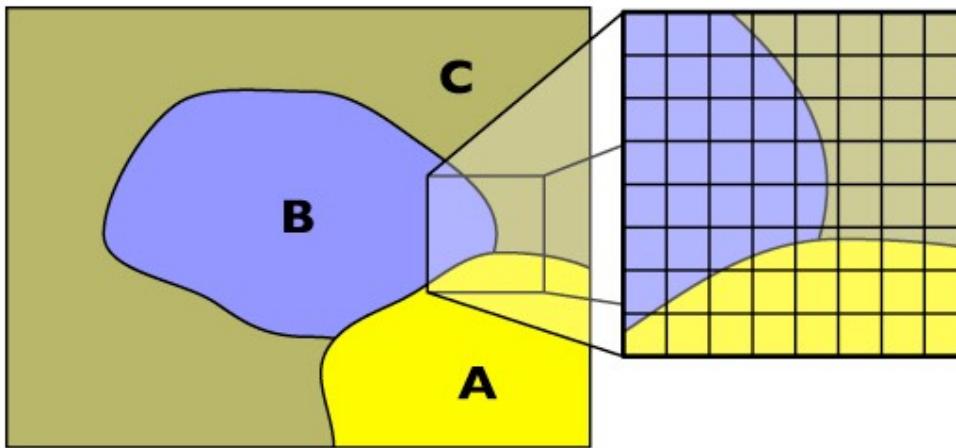


- in $D = A \text{ over } (B \text{ over } C)$ what will be the result?

$$\begin{aligned}c_D &= \alpha_A c_A + \bar{\alpha}_A [\alpha_B c_B + \bar{\alpha}_B c_C] \\&= \alpha_A c_A + \bar{\alpha}_A \alpha_B c_B + \bar{\alpha}_A \bar{\alpha}_B c_C\end{aligned}$$

Compositing composites

- Some pixels are partly covered in more than one layer

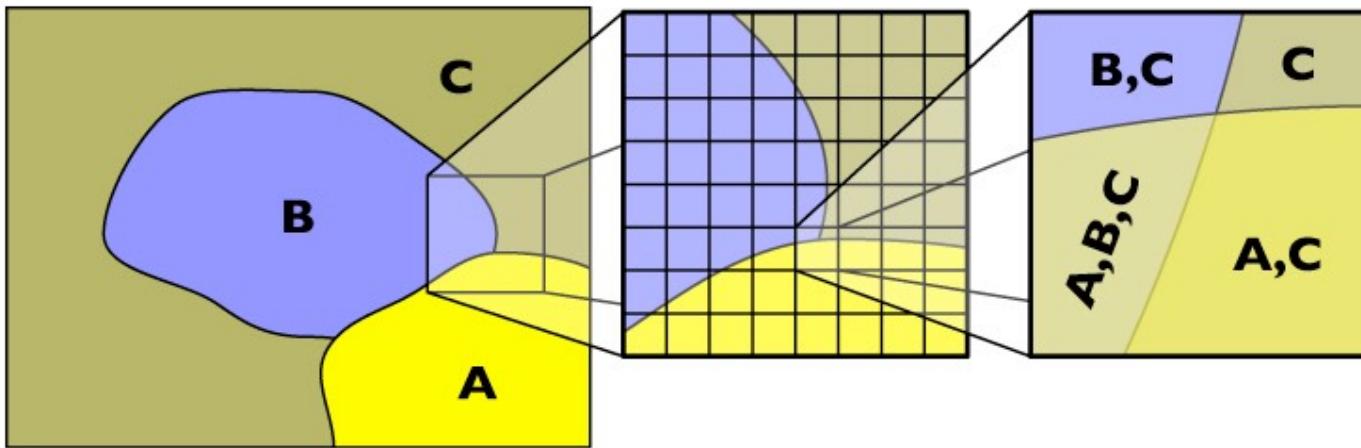


- in $D = A \text{ over } (B \text{ over } C)$ what will be the result?

$$\begin{aligned}c_D &= \alpha_A c_A + \bar{\alpha}_A [\alpha_B c_B + \bar{\alpha}_B c_C] \\&= \alpha_A c_A + \bar{\alpha}_A \alpha_B c_B + \bar{\alpha}_A \bar{\alpha}_B c_C\end{aligned}$$

Compositing composites

- Some pixels are partly covered in more than one layer

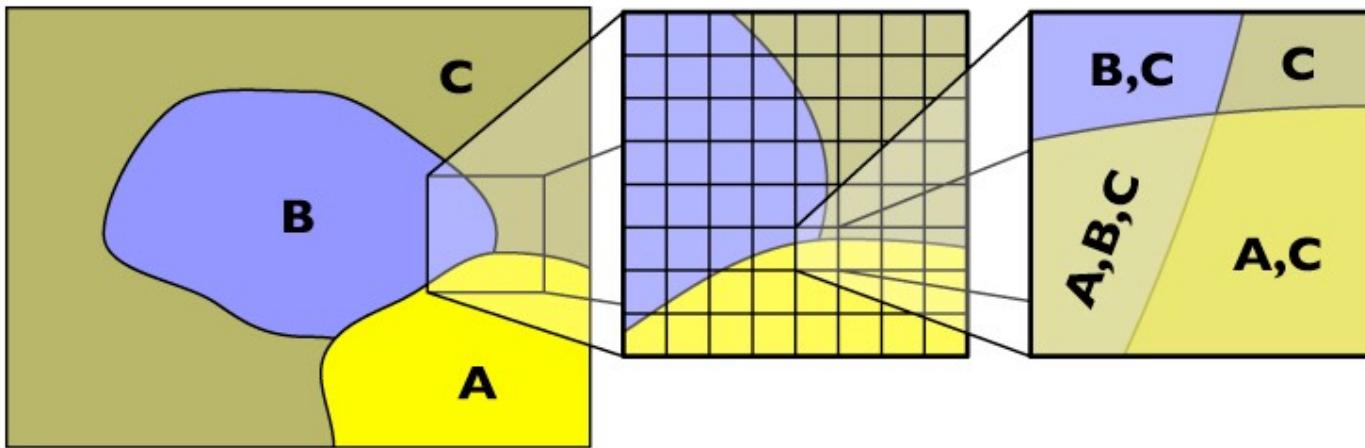


- in $D = A \text{ over } (B \text{ over } C)$ what will be the result?

$$\begin{aligned}c_D &= \alpha_A c_A + \bar{\alpha}_A [\alpha_B c_B + \bar{\alpha}_B c_C] \\&= \alpha_A c_A + \bar{\alpha}_A \alpha_B c_B + \bar{\alpha}_A \bar{\alpha}_B c_C\end{aligned}$$

Compositing composites

- Some pixels are partly covered in more than one layer



– in $D = A \text{ over } (B \text{ over } C)$ what will be the result?

$$\begin{aligned}c_D &= \alpha_A c_A + \bar{\alpha}_A [\alpha_B c_B + \bar{\alpha}_B c_C] \\&= \alpha_A c_A + \bar{\alpha}_A \alpha_B c_B + \bar{\alpha}_A \bar{\alpha}_B c_C\end{aligned}$$

Fraction covered by neither A nor B

Associativity?

- What does this imply about (A **over** B)?
 - Coverage has to be

$$\begin{aligned}\alpha_{(A \text{ over } B)} &= 1 - \bar{\alpha}_A \bar{\alpha}_B \\ &= \alpha_A + \bar{\alpha}_A \alpha_B = \alpha_B + \alpha_A \bar{\alpha}_B\end{aligned}$$

- ...but the color values then don't come out nicely in $D = (A \text{ over } B) \text{ over } C$, the color should be the same as in $A \text{ over } (B \text{ over } C)$:

$$\begin{aligned}c_D &= \alpha_{(A \text{ over } B)}(\dots) + \bar{\alpha}_{(A \text{ over } B)} c_C \\ &= (\alpha_A + \bar{\alpha}_A \alpha_B)(\dots) + \bar{\alpha}_A \bar{\alpha}_B c_C \\ &= \alpha_A c_A + \bar{\alpha}_A \alpha_B c_B + \bar{\alpha}_A \bar{\alpha}_B c_C\end{aligned}$$

An improvement

- Compositing equation again

$$c_E = \alpha_A c_A + (1 - \alpha_A) c_B$$

- Note c_A appears only in the product $\alpha_A c_A$
 - so why not do the multiplication ahead of time?
 - Leads to *premultiplied alpha*:
 - store pixel value (r', g', b', α) where $c' = \alpha c$
 - $E = A \text{ over } B$ becomes
- $$c'_E = c'_A + (1 - \alpha_A) c'_B$$
- this turns out to be more than an optimization...
 - hint: so far the background has been opaque!

Premultiplied alpha



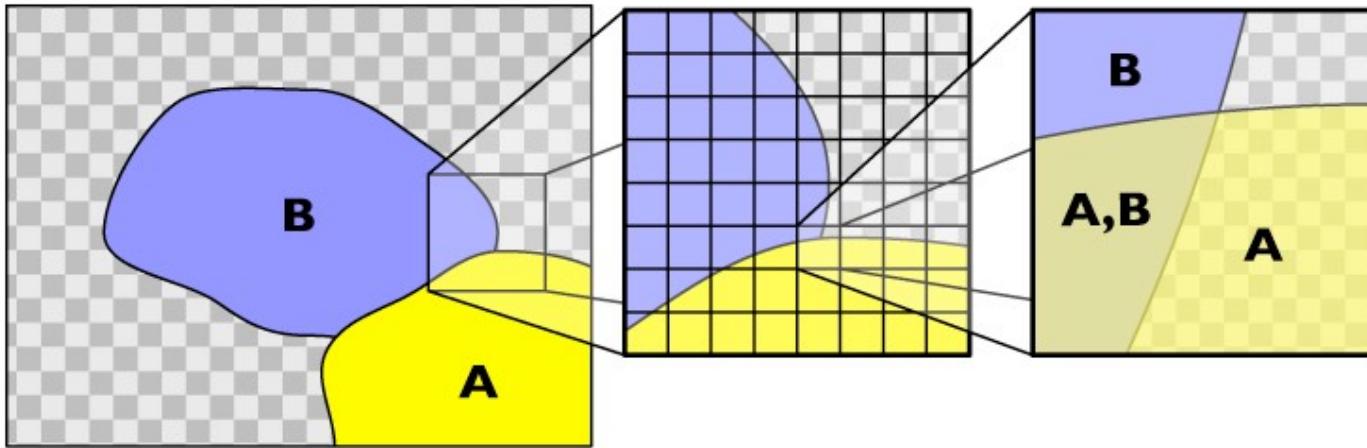
RGB



α

Compositing composites

- What about just $E = A \text{ over } B$ (with B transparent)?

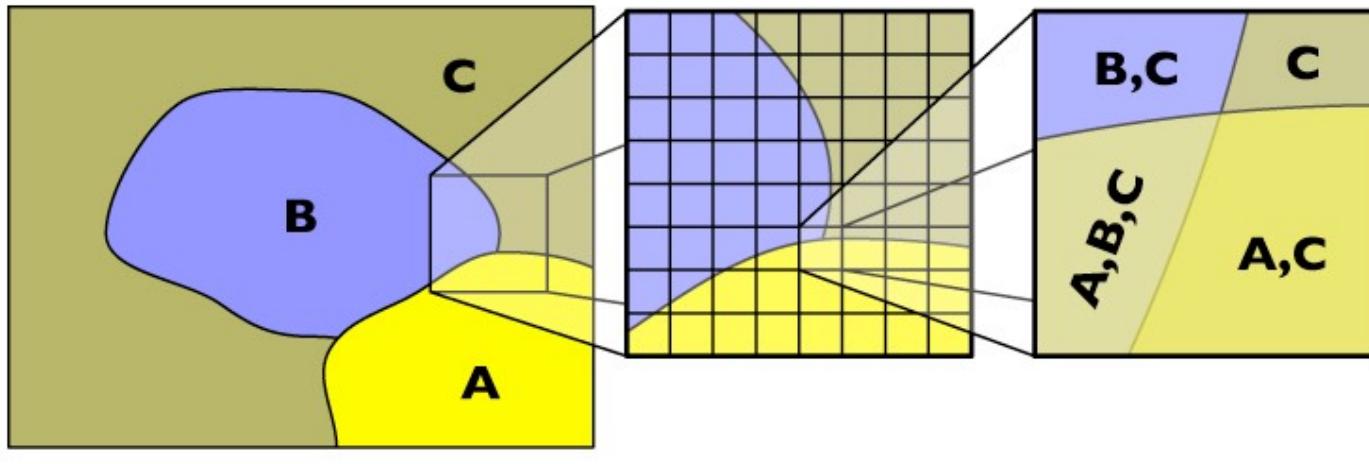


- in premultiplied alpha, the result

$$(c'_A, \alpha_A) \text{ over } (c'_B, \alpha_B) = (c'_A + \bar{\alpha}_A c_B, \alpha'_A + \bar{\alpha}_A \alpha_B)$$

treats alpha just like colors, and it leads to associativity.

Associativity!



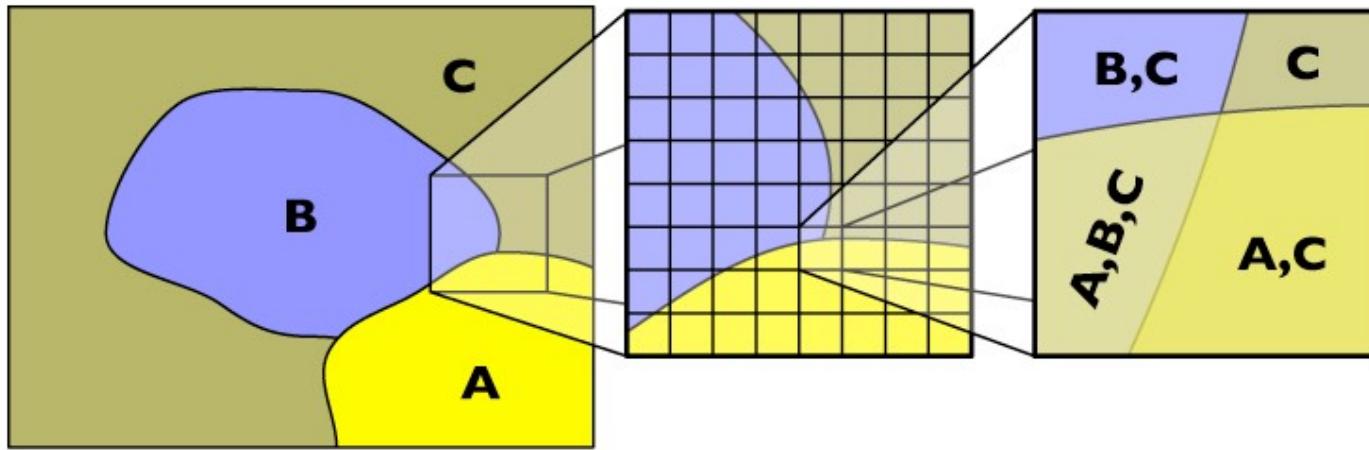
A over (B over C)

(A over B) over C

color

alpha

Associativity!



A over (B over C)

(A over B) over C

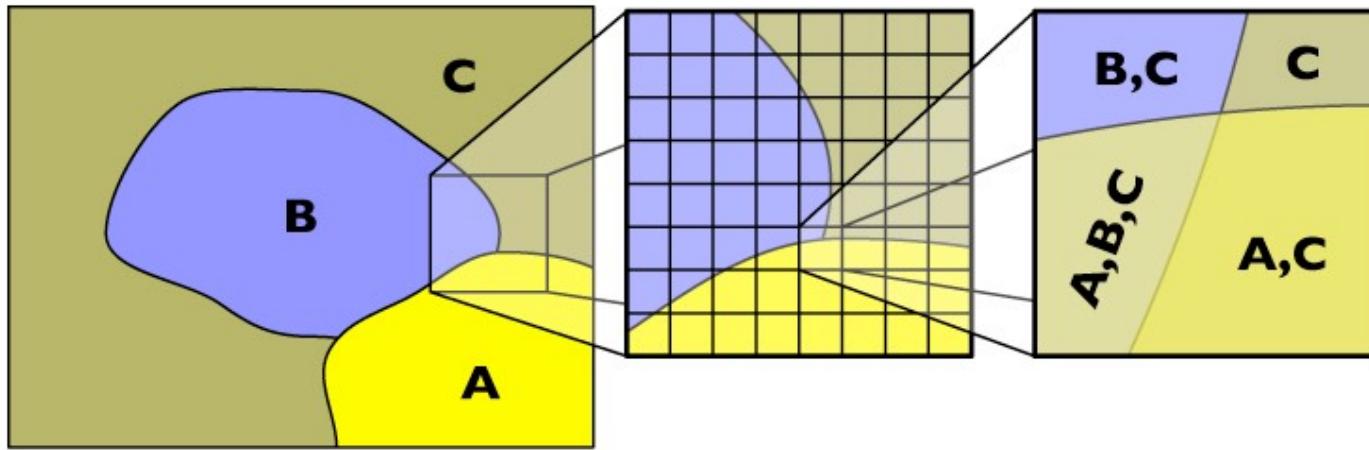
color

$$c'_A + \bar{\alpha}_A [c'_B + \bar{\alpha}_B c'_C]$$

$$c'_A + \bar{\alpha}_A c'_B + \bar{\alpha}_A \bar{\alpha}_B c'_C$$

alpha

Associativity!



A over (B over C)

color

$$c'_A + \bar{\alpha}_A [c'_B + \bar{\alpha}_B c'_C]$$

$$c'_A + \bar{\alpha}_A c'_B + \bar{\alpha}_A \bar{\alpha}_B c'_C$$

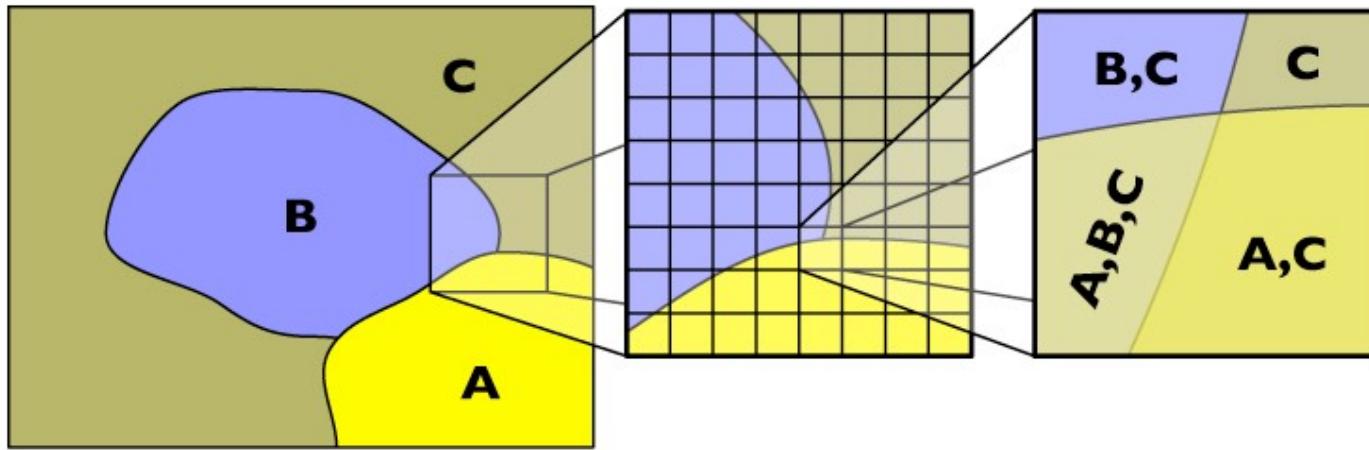
(A over B) over C

alpha

$$[c'_A + \bar{\alpha}_A c'_B] + \bar{\alpha}_A \bar{\alpha}_B c'_C$$

$$c'_A + \bar{\alpha}_A c'_B + \bar{\alpha}_A \bar{\alpha}_B c'_C$$

Associativity!



A over (B over C)

color

$$c'_A + \bar{\alpha}_A [c'_B + \bar{\alpha}_B c'_C]$$

$$c'_A + \bar{\alpha}_A c'_B + \bar{\alpha}_A \bar{\alpha}_B c'_C$$

(A over B) over C

alpha

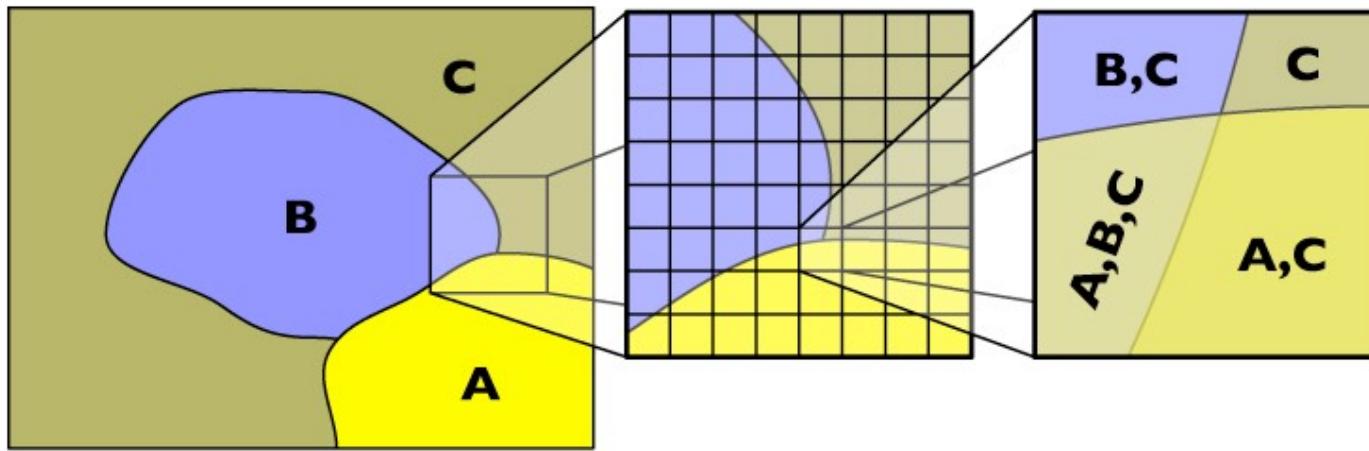
$$\alpha_A + \bar{\alpha}_A [\alpha_B + \bar{\alpha}_B \alpha_C]$$

$$\alpha_A + \bar{\alpha}_A \alpha_B + \bar{\alpha}_A \bar{\alpha}_B \alpha_C$$

$$[c'_A + \bar{\alpha}_A c'_B] + \bar{\alpha}_A \bar{\alpha}_B c'_C$$

$$c'_A + \bar{\alpha}_A c'_B + \bar{\alpha}_A \bar{\alpha}_B c'_C$$

Associativity!



A over (B over C)

color

$$c'_A + \bar{\alpha}_A [c'_B + \bar{\alpha}_B c'_C]$$

$$c'_A + \bar{\alpha}_A c'_B + \bar{\alpha}_A \bar{\alpha}_B c'_C$$

alpha

$$\alpha_A + \bar{\alpha}_A [\alpha_B + \bar{\alpha}_B \alpha_C]$$

$$\alpha_A + \bar{\alpha}_A \alpha_B + \bar{\alpha}_A \bar{\alpha}_B \alpha_C$$

(A over B) over C

$$[c'_A + \bar{\alpha}_A c'_B] + \bar{\alpha}_A \bar{\alpha}_B c'_C$$

$$c'_A + \bar{\alpha}_A c'_B + \bar{\alpha}_A \bar{\alpha}_B c'_C$$

$$[\alpha_A + \bar{\alpha}_A \alpha_B] + \bar{\alpha}_A \bar{\alpha}_B \alpha_C$$

$$\alpha_A + \bar{\alpha}_A \alpha_B + \bar{\alpha}_A \bar{\alpha}_B \alpha_C$$

Compositing algebras

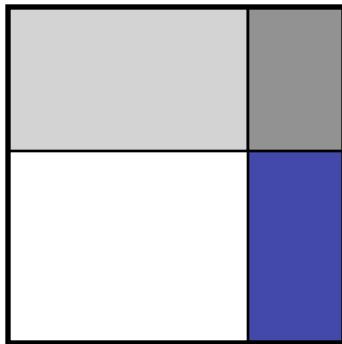
- Compositing is a multiply-and-add operation
 - “take the color you have, scale it by 1-alpha, add some more”
- This is a 1D affine transformation

$$\begin{bmatrix} \bar{\alpha}_A & c'_A \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \bar{\alpha}_B & c'_B \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \bar{\alpha}_A \bar{\alpha}_B & c'_A + \bar{\alpha}_A c'_B \\ 0 & 1 \end{bmatrix}$$

- Once we write it like this, associativity is automatically inherited from matrix multiplication!
 - and we realize we haven’t invented any new math after all.

Independent coverage assumption

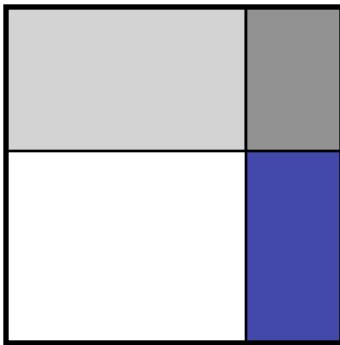
- Why is it reasonable to blend α like a color?
- Simplifying assumption: covered areas are independent
 - that is, uncorrelated in the statistical sense



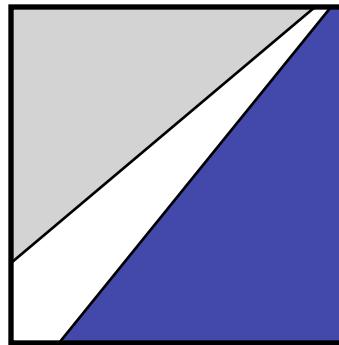
<i>description</i>	<i>area</i>
$\bar{A} \cap \bar{B}$	$(1-\alpha_A)(1-\alpha_B)$
$A \cap \bar{B}$	$\alpha_A(1-\alpha_B)$
$\bar{A} \cap B$	$(1-\alpha_A)\alpha_B$
$A \cap B$	$\alpha_A\alpha_B$

Independent coverage assumption

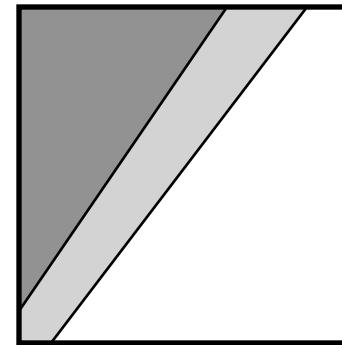
- Holds in most but not all cases



this



not this



or this

- This will cause artifacts
 - but we'll carry on anyway because it is simple and usually works...

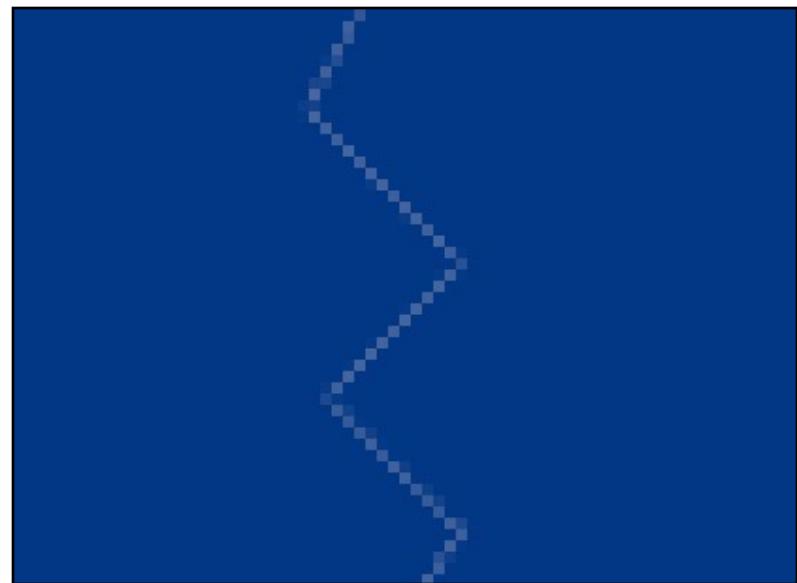
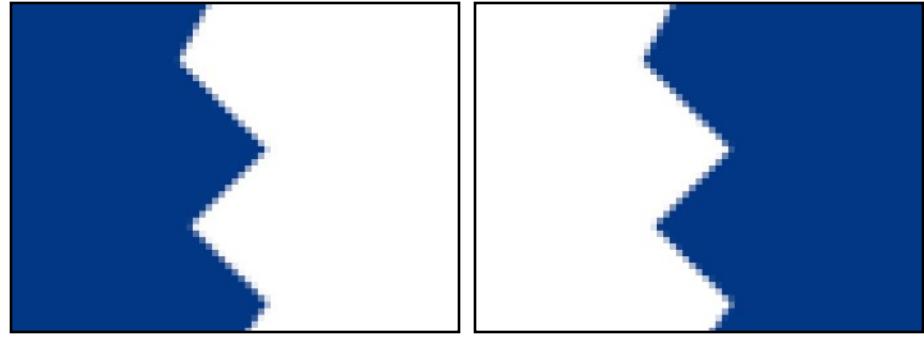
Alpha compositing—failures

[Cornell PCG]



[Chuang et al. / Core]

positive correlation:
too much foreground



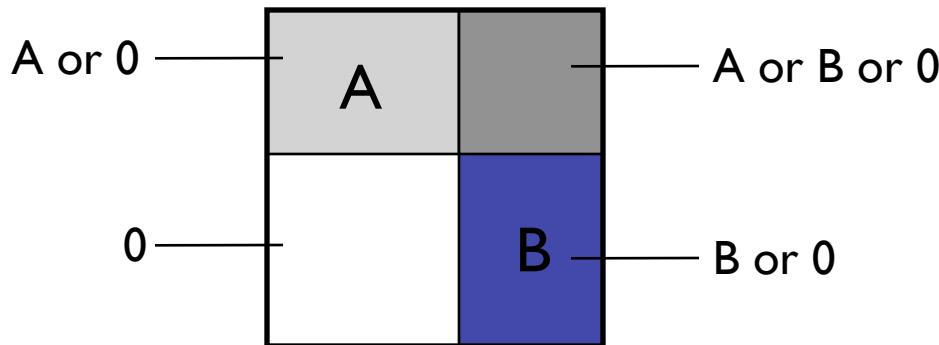
negative correlation:
too little foreground

Other compositing operations

- Generalized form of compositing equation:

$$\alpha_E = A \text{ op } B$$

$$c'_E = F_A c'_A + F_B c'_B$$



$1 \times 2 \times 3 \times 2 = 12$ reasonable choices

operation	quadruple	diagram	F_A	F_B
clear	(0,0,0,0)		0	0
A	(0,A,0,A)		1	0
B	(0,0,B,B)		0	1
$A \text{ over } B$	(0,A,B,A)		1	$1-\alpha_A$
$B \text{ over } A$	(0,A,B,B)		$1-\alpha_B$	1
$A \text{ in } B$	(0,0,0,A)		α_B	0
$B \text{ in } A$	(0,0,0,B)		0	α_A
$A \text{ out } B$	(0,A,0,0)		$1-\alpha_B$	0
$B \text{ out } A$	(0,0,B,0)		0	$1-\alpha_A$
$A \text{ atop } B$	(0,0,B,A)		α_B	$1-\alpha_A$
$B \text{ atop } A$	(0,A,0,B)		$1-\alpha_B$	α_A
$A \text{ xor } B$	(0,A,B,0)		$1-\alpha_B$	$1-\alpha_A$