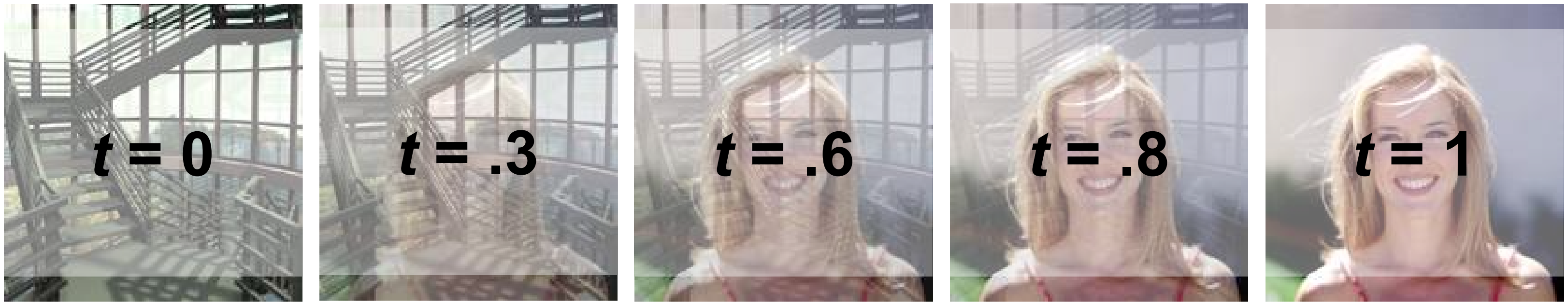
Compositing

[*Titanic* ; DigitalDomain; vfxhq.com]



Combining images

- Trivial example: video crossfade
  – smooth transition from one scene to another

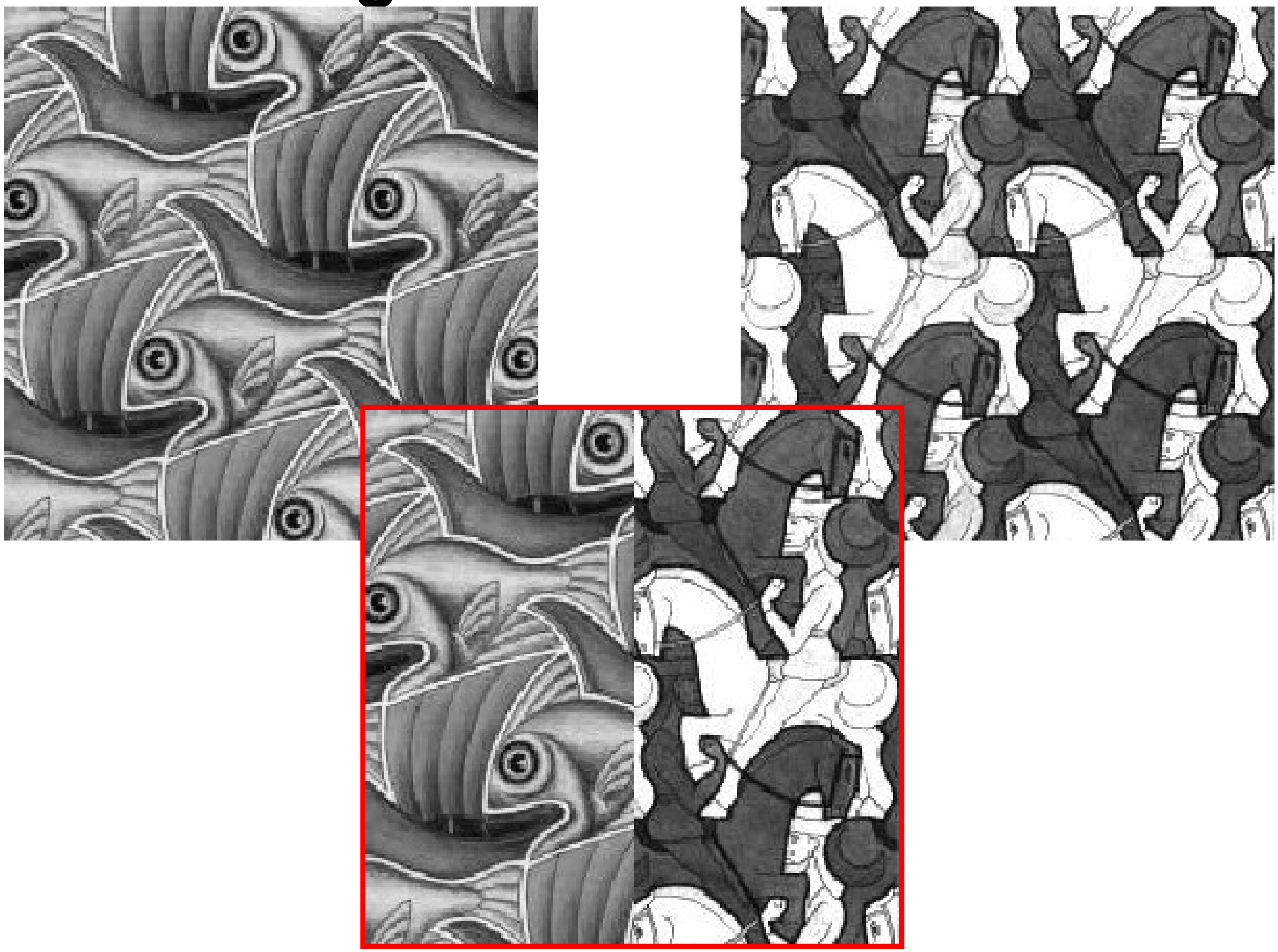
[Chuang et al. / Corel; Conell PCG]

$$r_C = t r_A + (1 - t) r_B$$
$$g_C = t g_A + (1 - t) g_B$$
$$b_C = t b_A + (1 - t) b_B$$

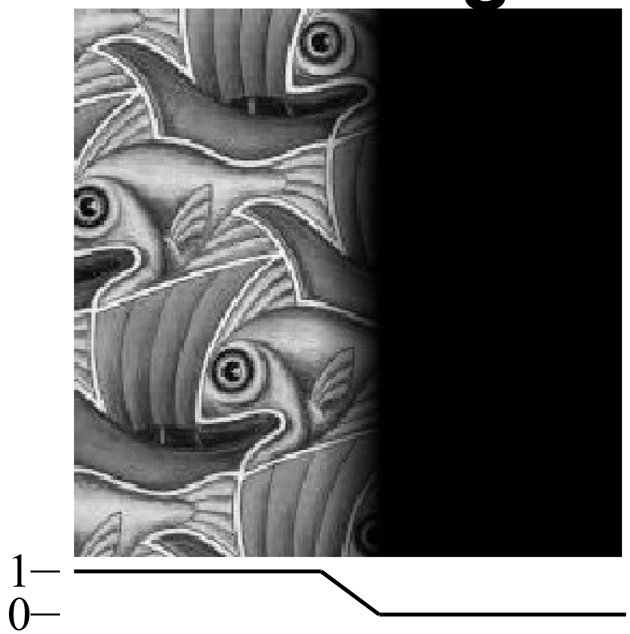
  – note: weights sum to 1.0
    - no unexpected brightening or darkening
    - no out-of-range results

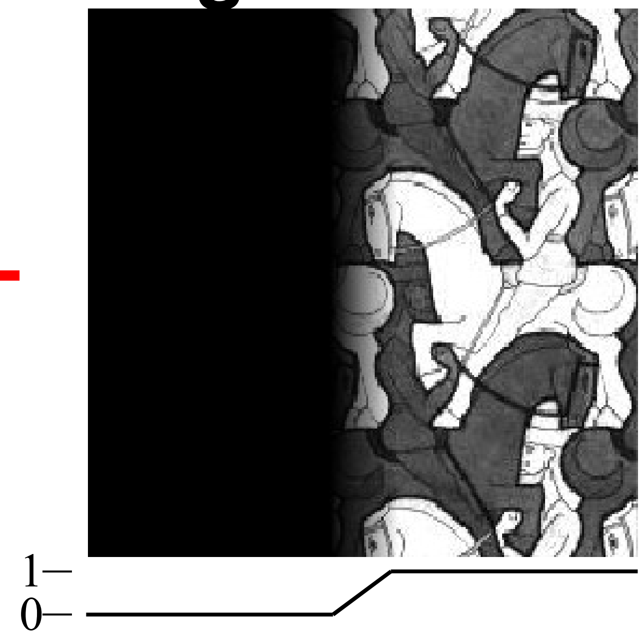
  – this is *linear interpolation*

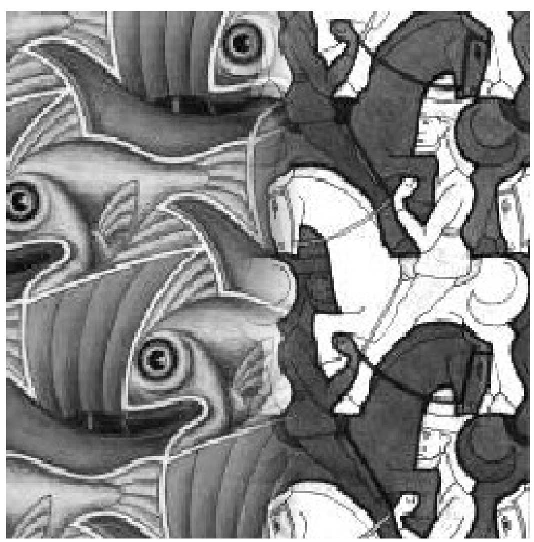
Need blending



Alpha Blending / Feathering







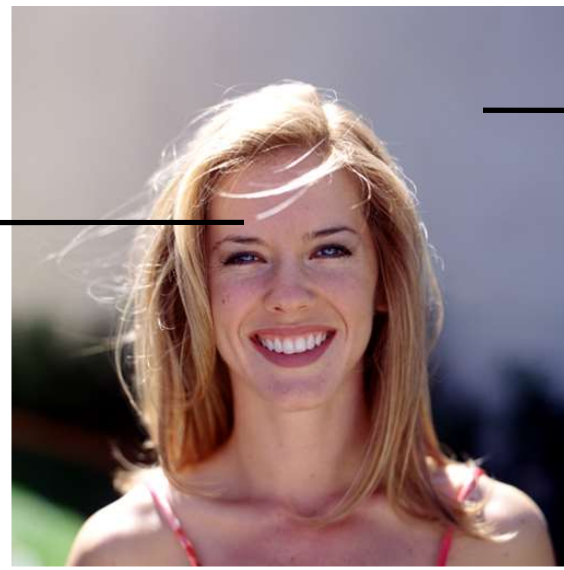
$$I_{blend} = \alpha I_{left} + (1-\alpha) I_{right}$$

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



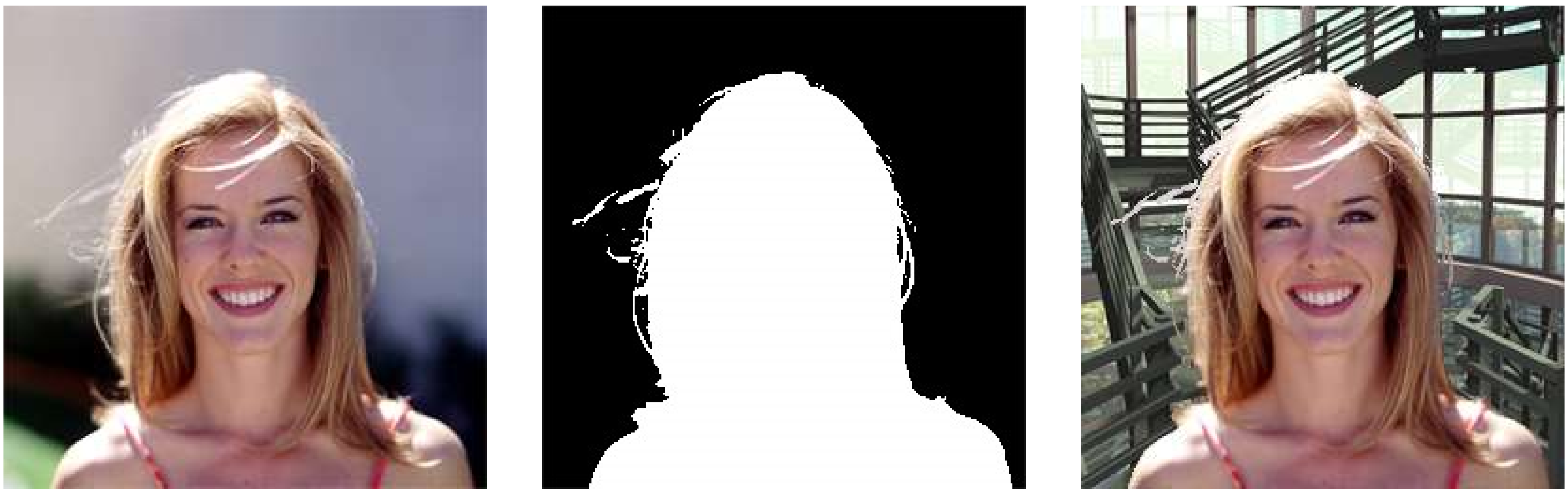
[Chuang et al. / Corel]

- 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?”

[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]

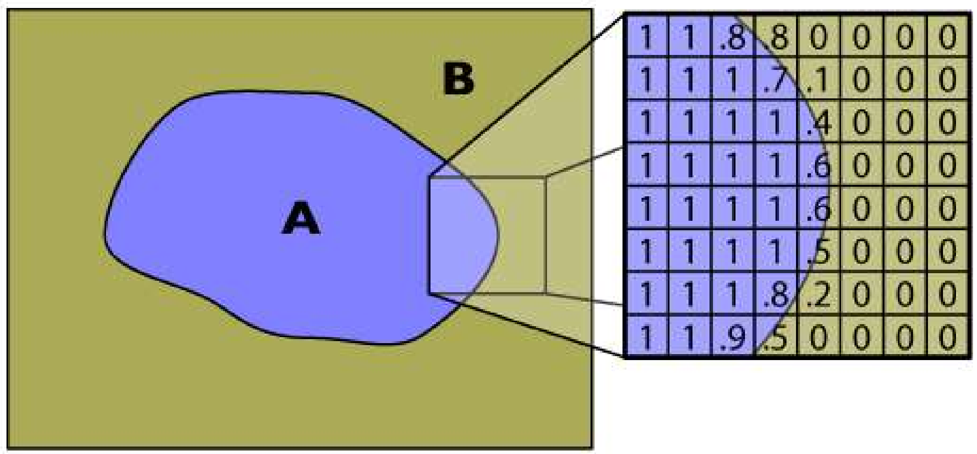
  - causes jaggies similar to point-sampled rasterization
  - same problem, same solution: intermediate values

Alpha compositing—example

[Chuang et al. / Corel]

Partial pixel coverage

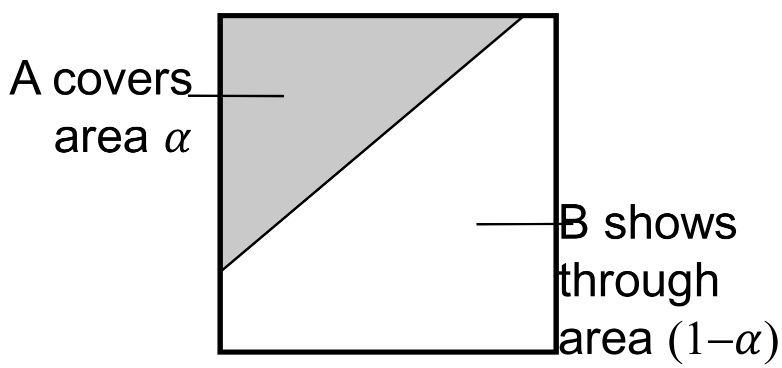
- The problem: pixels near boundary are not strictly foreground or background



  - 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



$$C = A \textbf{ over } B$$

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

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

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

  – this exactly like a spatially varying crossfade

- Convenient implementation
  – 8 more bits makes 32
  – 2 multiplies + 1 add per pixel for compositing

Alpha compositing—example

[Chuang et al. / Corel]

Compositing composites

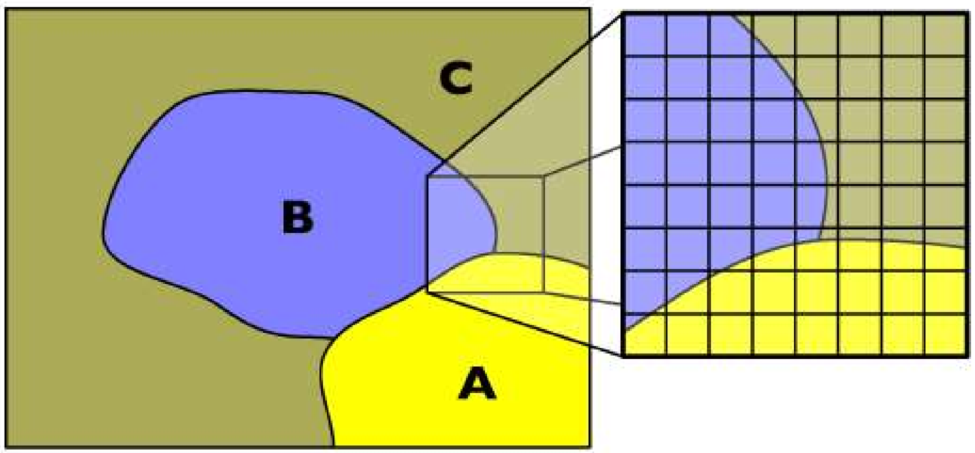
- so far have only considered single fg. over single bg.
- in real applications we have *n* layers
  - *Titanic* example
  - compositing foregrounds to create new foregrounds
    - what to do with α?
- desirable property: associativity

$$A \textbf{ over } (B \textbf{ over } C) = (A \textbf{ over } B) \textbf{ over } C$$

  - to make this work we need to be careful about how $\langle$ is computed

Compositing composites

- Some pixels are partly covered in more than one layer



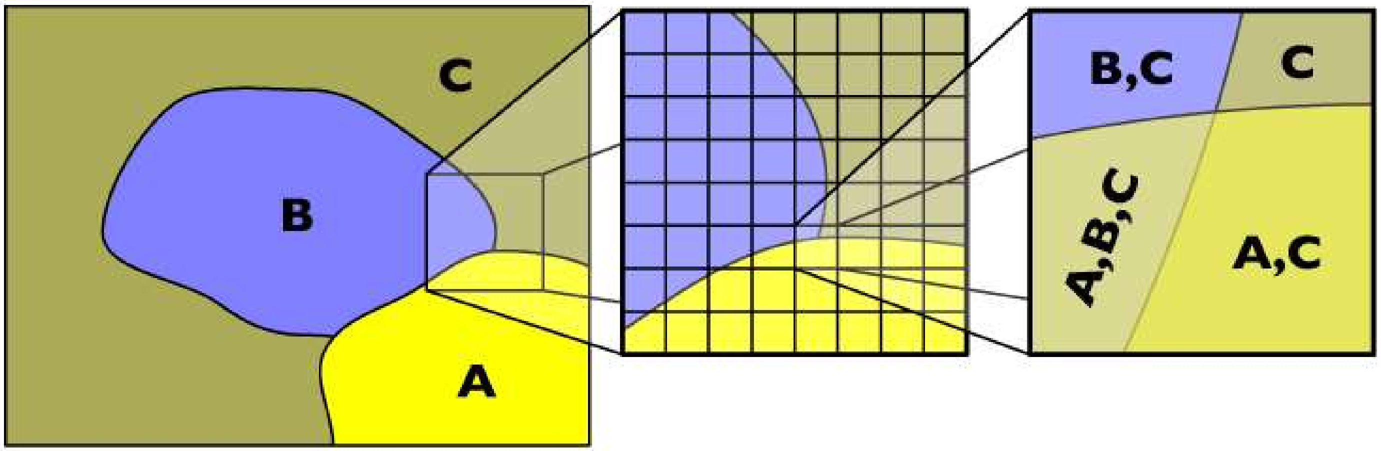
– in D = A **over** (B **over** C) what will be the result?

$$c_D = \alpha_A c_A + (1 - \alpha_A)[\alpha_B c_B + (1 - \alpha_B) c_C]$$
$$= \alpha_A c_A + (1 - \alpha_A)\alpha_B c_B + (1 - \alpha_A)(1 - \alpha_B) c_C$$

Fraction covered by neither A nor B

Compositing composites

- Some pixels are partly covered in more than one layer



– in D = A **over** (B **over** C) what will be the result?

$$c_D = \alpha_A c_A + (1 - \alpha_A)[\alpha_B c_B + (1 - \alpha_B) c_C]$$
$$= \alpha_A c_A + (1 - \alpha_A)\alpha_B c_B + (1 - \alpha_A)(1 - \alpha_B) c_C$$

Fraction covered by neither A nor B

Associativity?

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

$$\alpha_{(A \textbf{ over } B)} = 1 - (1 - \alpha_A)(1 - \alpha_B)$$
$$= \alpha_A + (1 - \alpha_A)\alpha_B$$

  – …but the color values then don't come out nicely in D = (A **over** B) **over** C:

$$c_D = \alpha_A c_A + (1 - \alpha_A)\alpha_B c_B + (1 - \alpha_A)(1 - \alpha_B)c_C$$
$$= \alpha_{(A \textbf{ over } B)}(\cdots) + (1 - \alpha_{(A \textbf{ over } B)})c_C$$

An optimization

- Compositing equation again

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

- This equation is correct only when the background is opaque!
- Otherwise,

$$C_C = \alpha_A C_A + (1 - \alpha_A) \alpha_B C_B$$

An optimization

- New compositing equation

$$C_C = \alpha_A C_A + (1 - \alpha_A)\alpha_B C_B$$

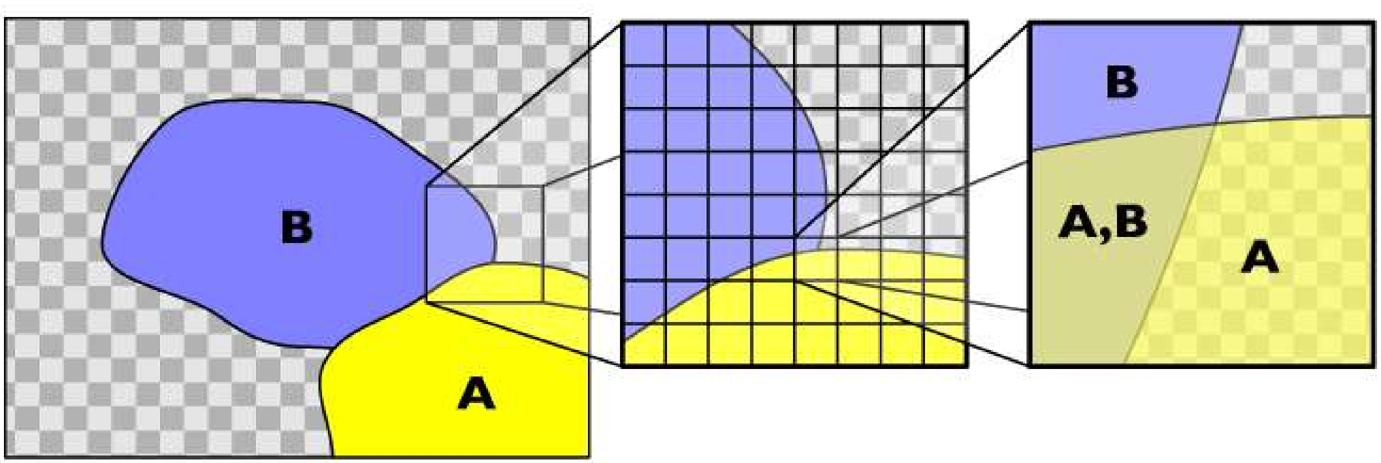
- Note c_A appears only in the product $\alpha_A c_A$
  - so why not do the multiplication ahead of time?
- Leads to *premultiplied alpha*:
  - store pixel value (r', g', b', α') where $c' = \alpha\, c$
  - C = A **over** B becomes

$$c'_C = c'_A + (1 - \alpha_A) c'_B$$

  - this turns out to be more than an optimization…

Compositing composites

- What about just C = A **over** B (with B transparent)?

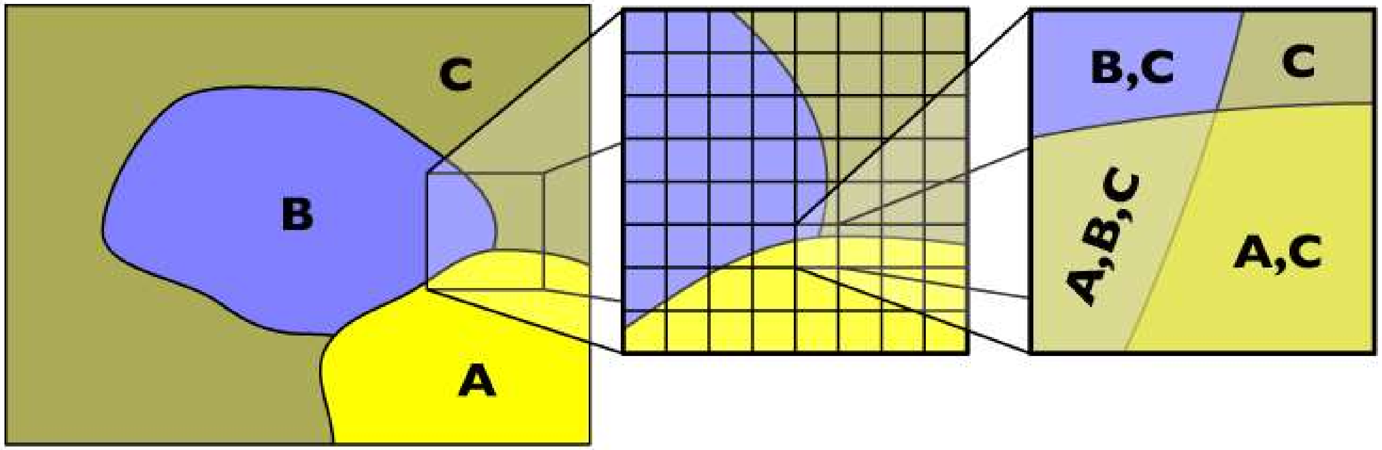


  – in premultiplied alpha, the result

$$\alpha_C = \alpha_A + (1 - \alpha_A)\alpha_B$$

  looks just like blending colors, and it leads to associativity.

Associativity!



$$c_D = c'_A + (1 - \alpha_A)[c'_B + (1 - \alpha_B)c'_C]$$
$$= [c'_A + (1 - \alpha_A)c'_B] + (1 - \alpha_A)(1 - \alpha_B)c'_C$$
$$= c'_{(A \textbf{ over } B)} + (1 - \alpha_{(A \textbf{ over } B)})c'_C$$

– This is another good reason to premultiply

Summary

- A over B:

$$c'_C = c'_A + (1 - \alpha_A) c'_B$$
$$\alpha_C = \alpha_A + (1 - \alpha_A) \alpha_B$$

c_A

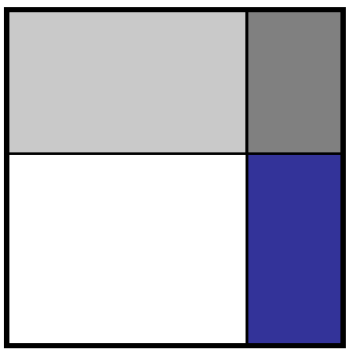
α_A

c'_A

?

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

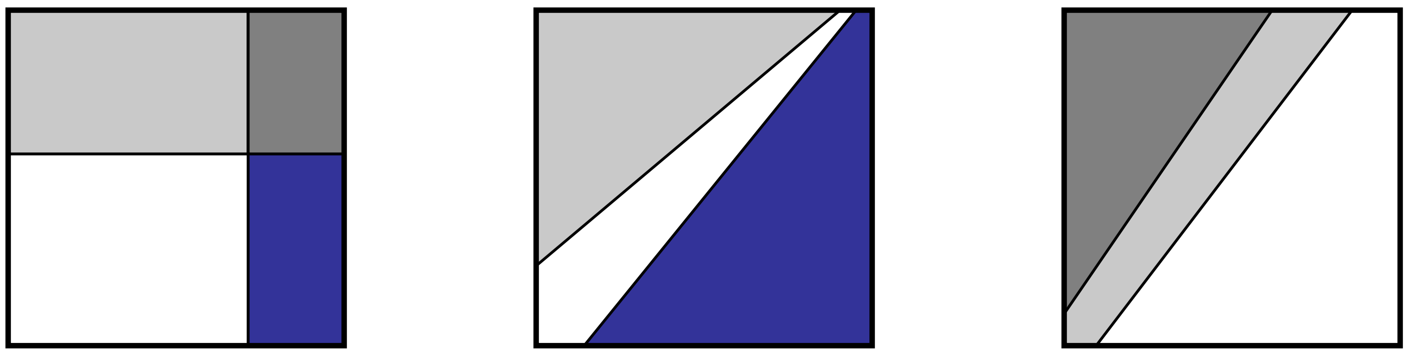


description	area
$\bar{A} \cap \bar{B}$	$(1-\alpha_A)(1-\alpha_B)$
$A \cap \bar{B}$	$\alpha_A(1-\alpha_B)$
$\bar{A} \cap B$	$(1-\alpha_A)\alpha_B$
$A \cap B$	$\alpha_A \alpha_B$

[Porter & Duff 84]

Independent coverage assumption

- Holds in most but not all cases



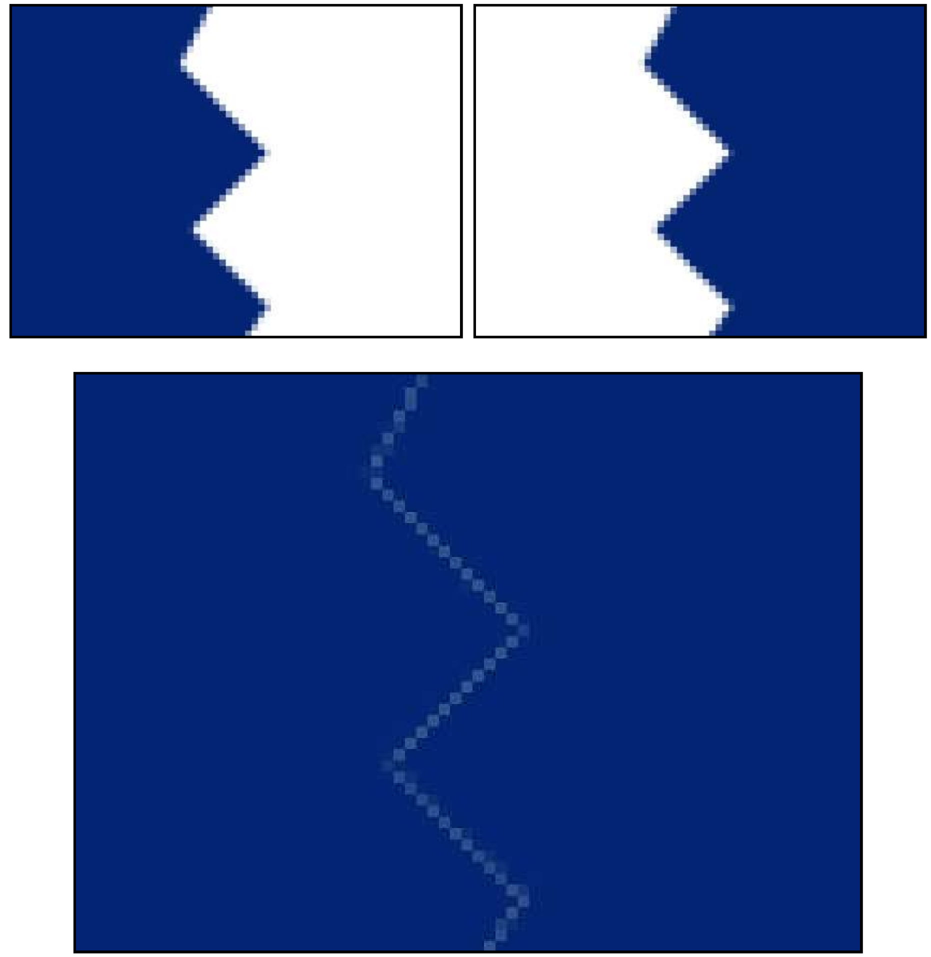
this | not this | or this

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

Alpha compositing—failures

[Chuang et al. / Corel] [Cornell PCG]

positive correlation:
too much foreground



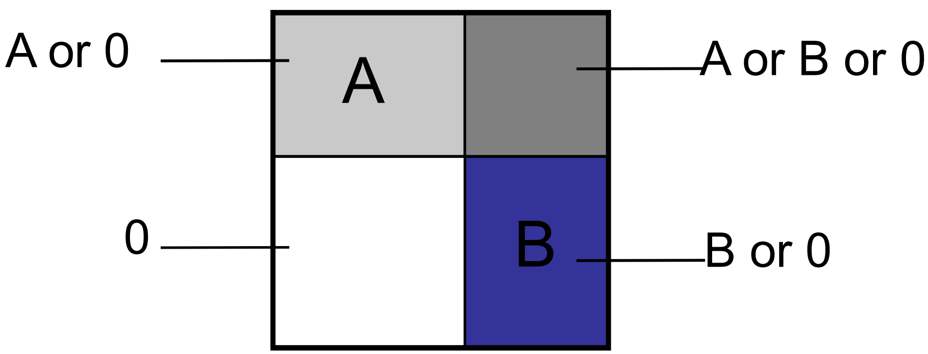
negative correlation:
too little foreground

Other compositing operations

- Generalized form of compositing equation:

$$\alpha C = A \ \mathbf{op} \ B$$

$$c = F_A a + F_B b$$

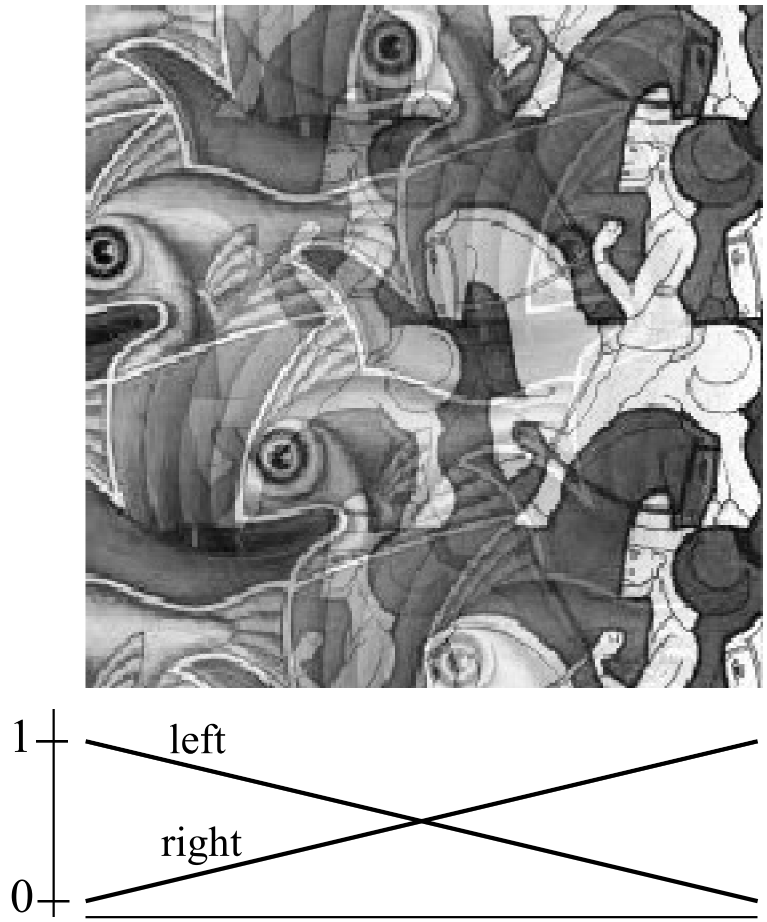


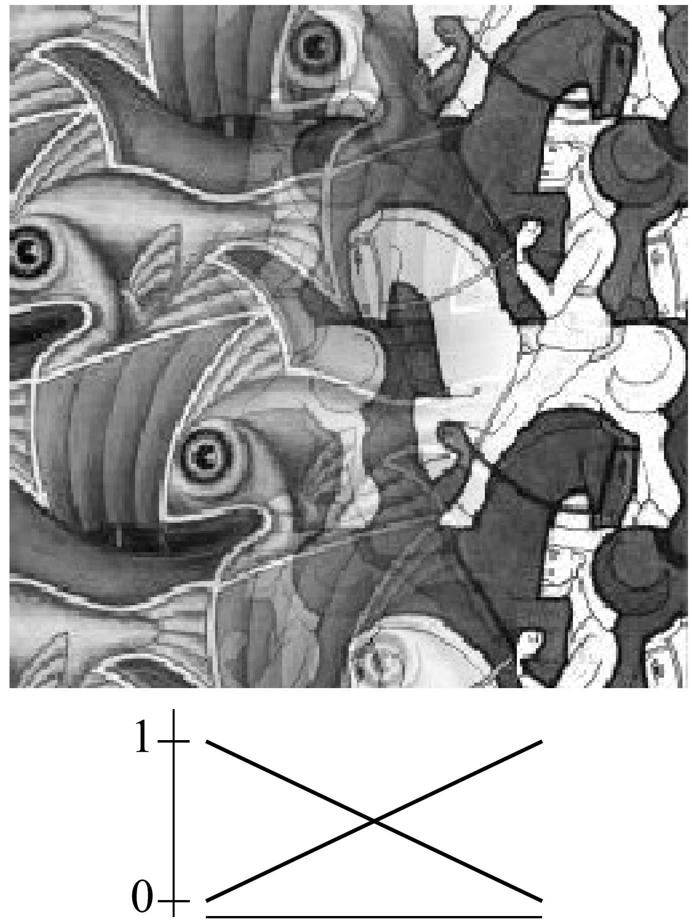
1 x 2 x 3 x 2 = 12 reasonable choices

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 **over** B	(0,A,B,A)		1	$1-\alpha_A$
B **over** A	(0,A,B,B)		$1-\alpha_B$	1
A **in** B	(0,0,0,A)		α_B	0
B **in** A	(0,0,0,B)		0	α_A
A **out** B	(0,A,0,0)		$1-\alpha_B$	0
B **out** A	(0,0,B,0)		0	$1-\alpha_A$
A **atop** B	(0,0,B,A)		α_B	$1-\alpha_A$
B **atop** A	(0,A,0,B)		$1-\alpha_B$	α_A
A **xor** B	(0,A,B,0)		$1-\alpha_B$	$1-\alpha_A$

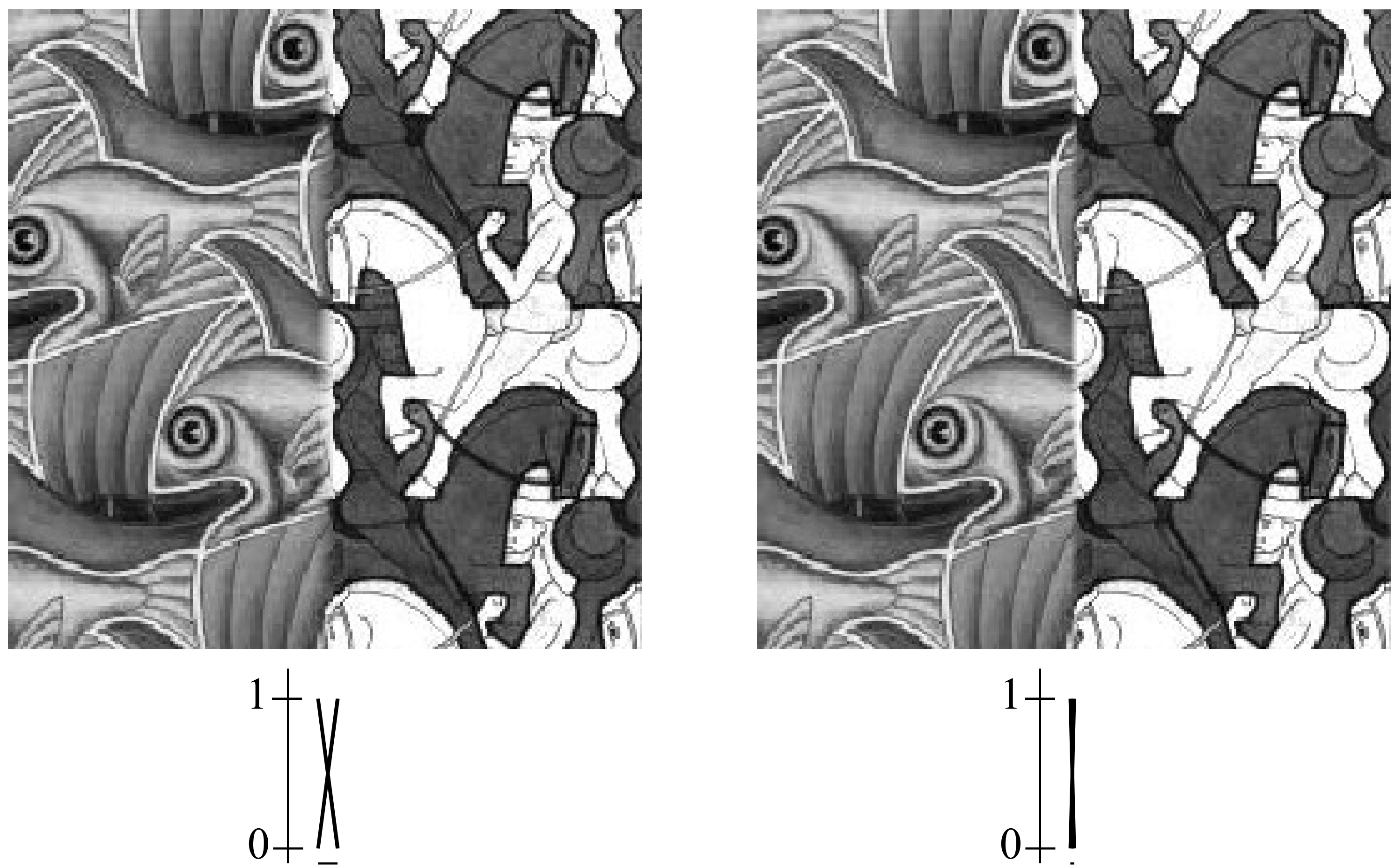
[Porter & Duff 84]

Affect of Window Size

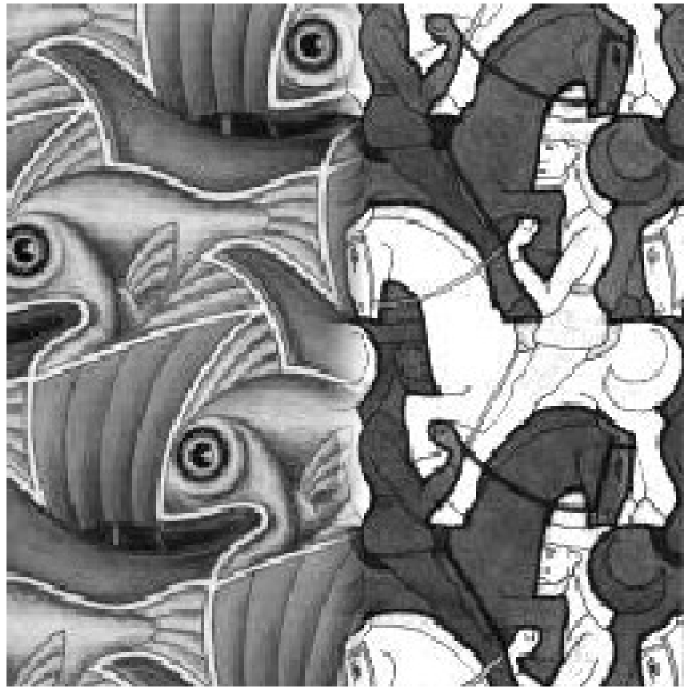


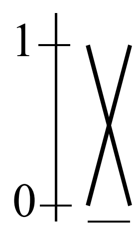


Affect of Window Size



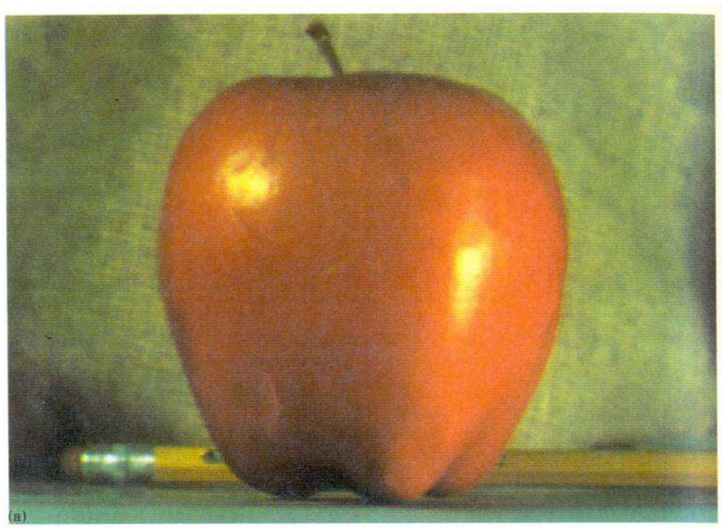
Good Window Size

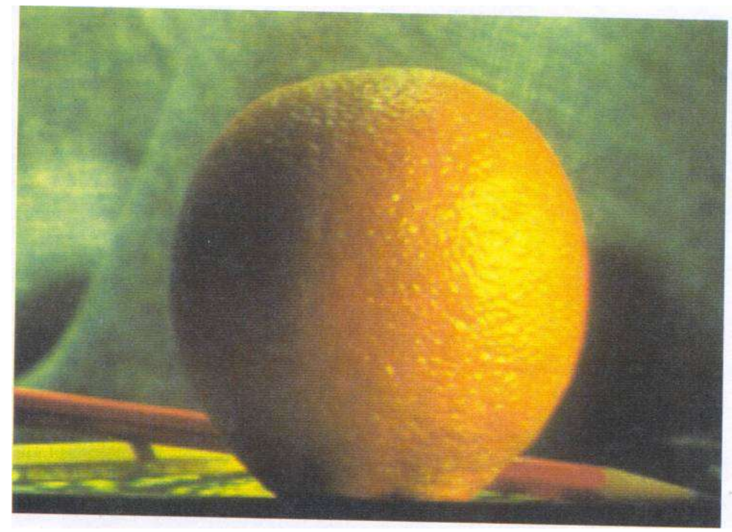


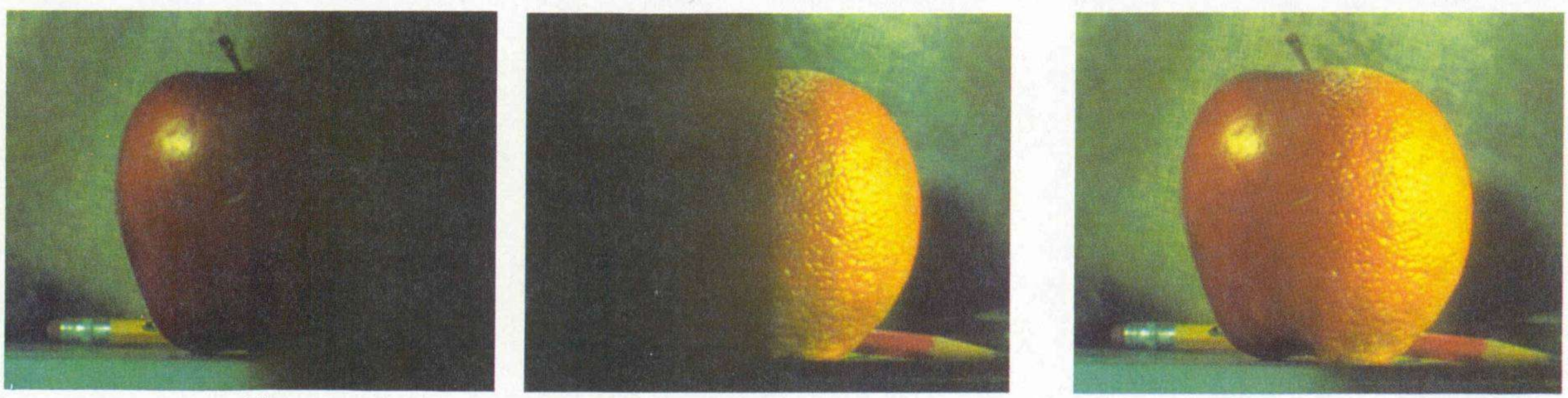


"Optimal" Window: smooth but not ghosted

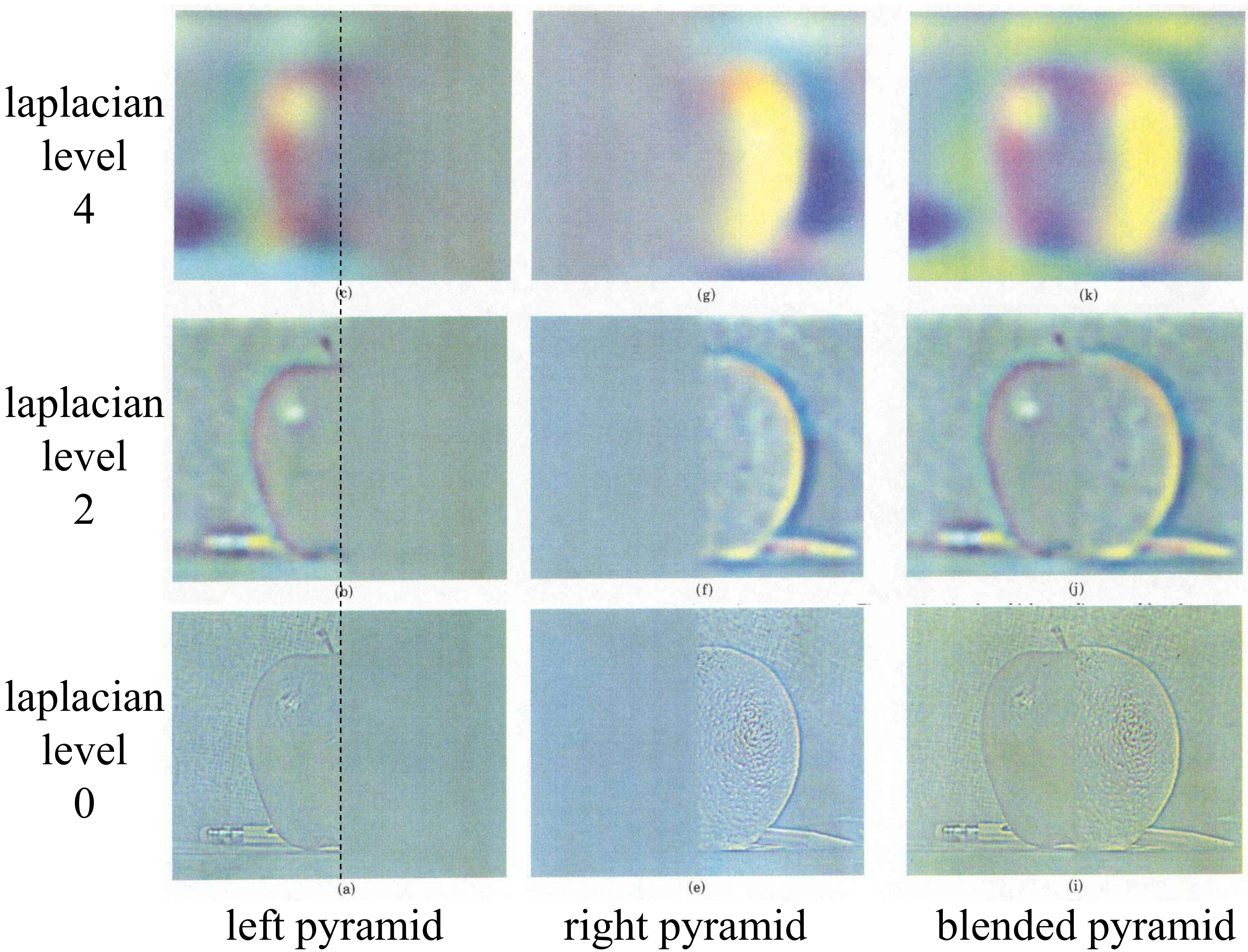
Pyramid Blending







(d) (h) (l)


laplacian level 4
laplacian level 2
laplacian level 0
(c)
(g)
(k)
(b)
(f)
(j)
(a)
(e)
(i)
left pyramid
right pyramid
blended pyramid

Simplification: Two-band Blending

Brown & Lowe, 2003

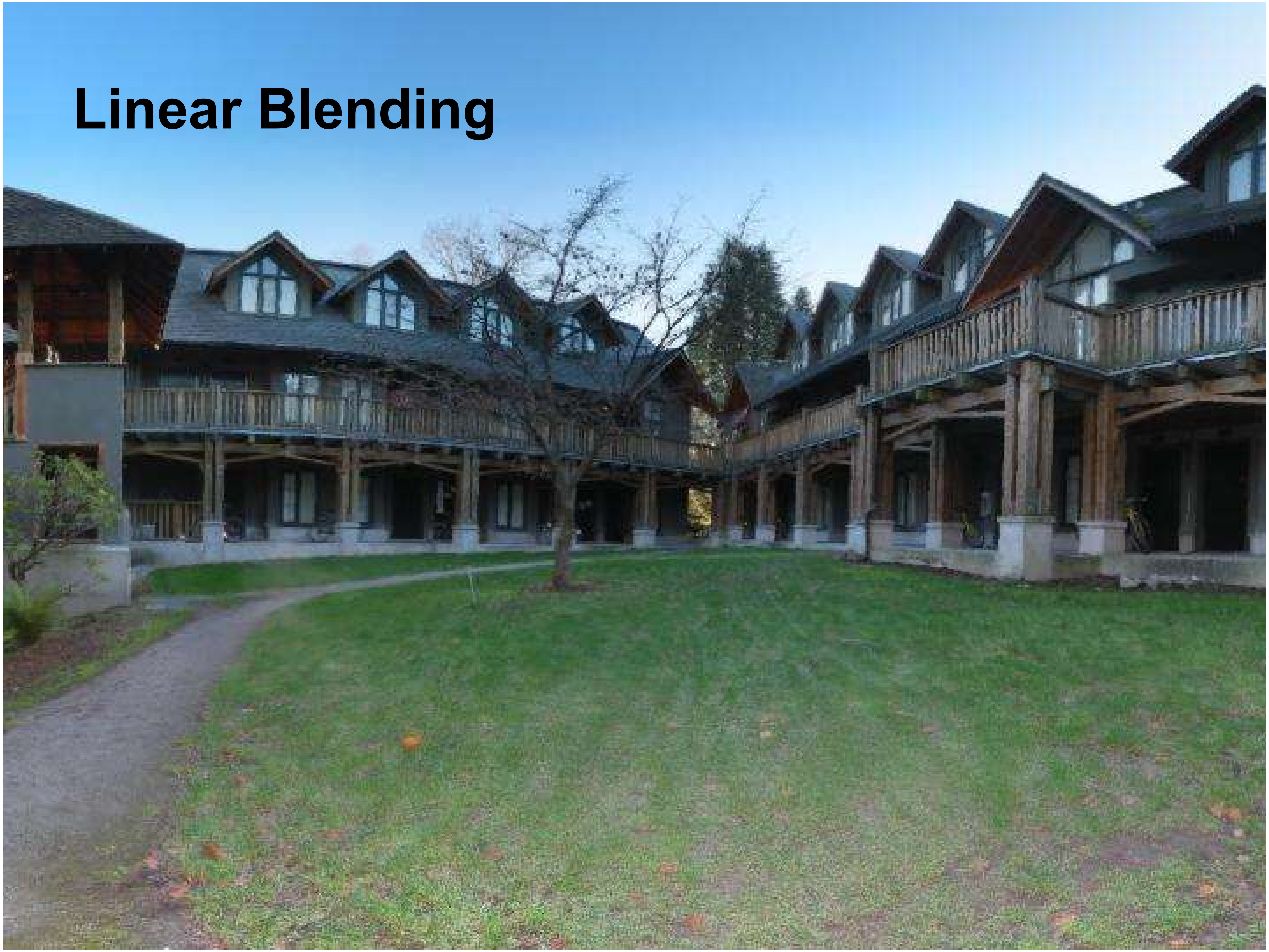
- Only use two bands: high freq. and low freq.
- Blends low freq. smoothly
- Blend high freq. with no smoothing: use binary alpha

2-band Blending

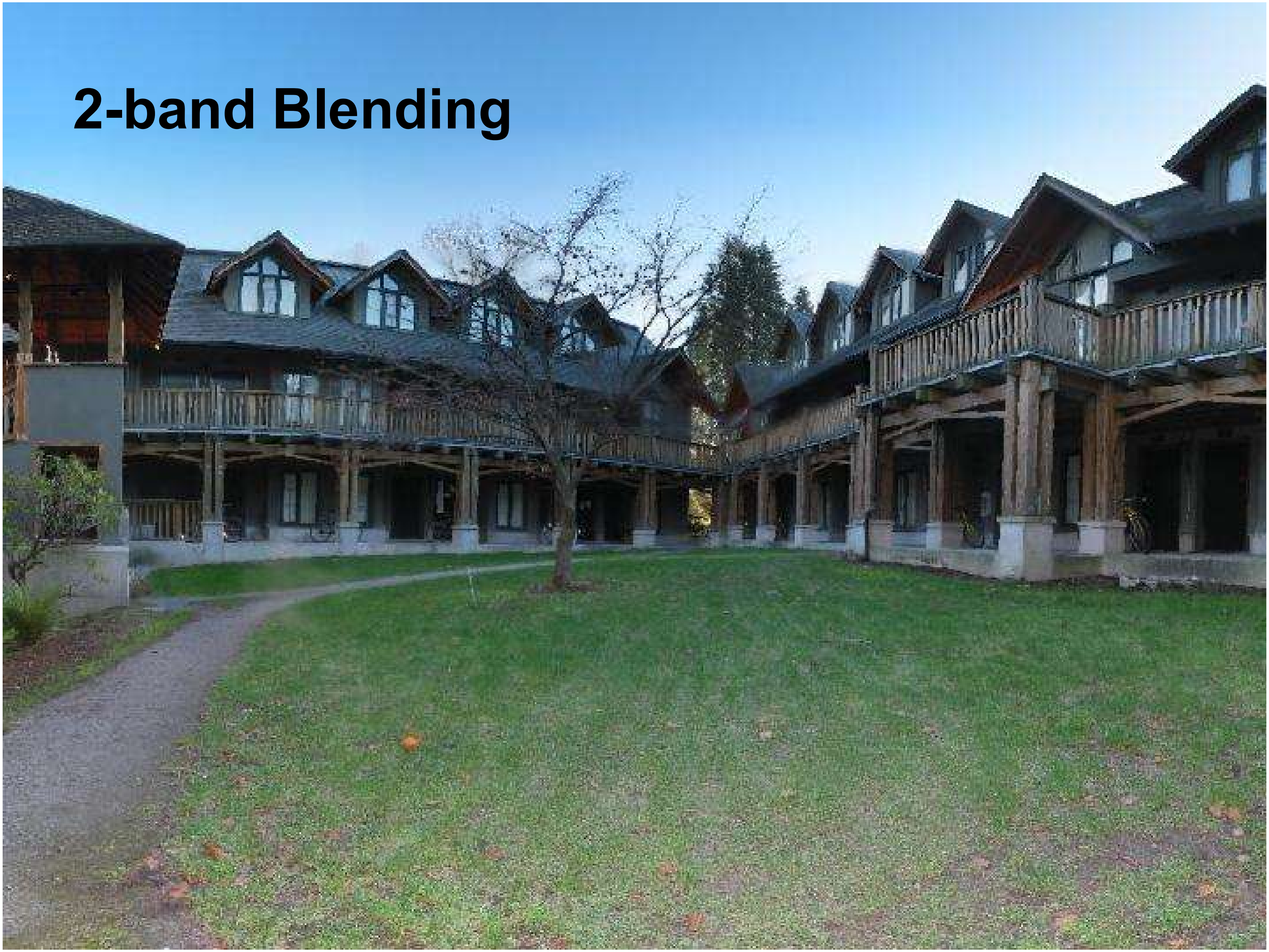
Low frequency (λ > 2 pixels)

High frequency (λ < 2 pixels)

Linear Blending



2-band Blending



Don’t blend, CUT!

Moving objects become ghosts

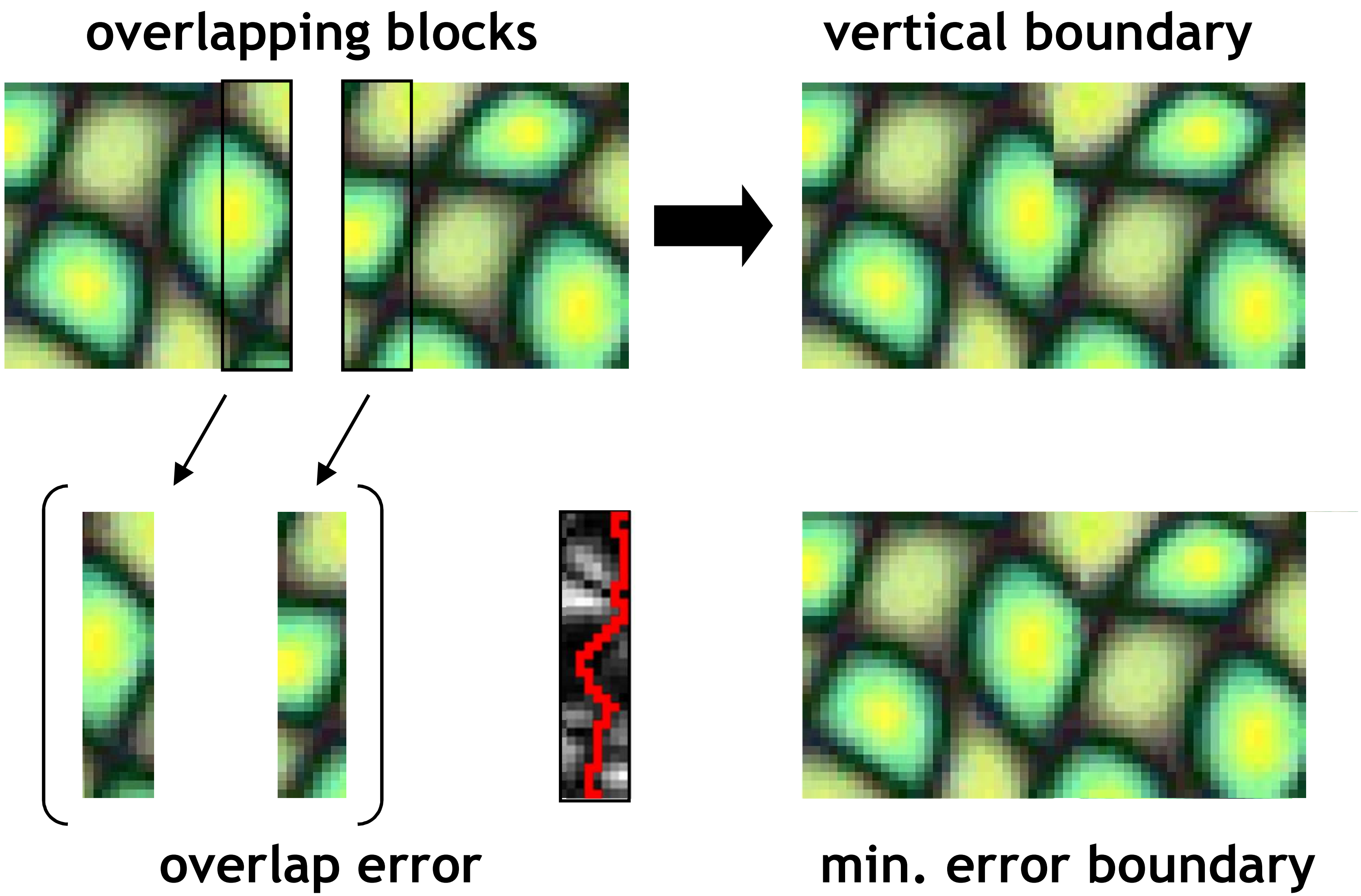
So far we only tried to blend between two images. What about finding an optimal seam?

Davis, 1998

Segment the mosaic

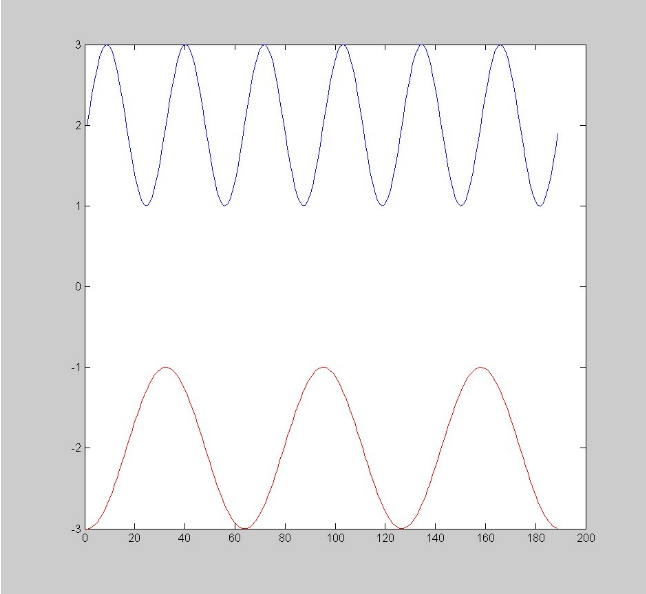
- Single source image per segment
- Avoid artifacts along boundries
  - Dijkstra’s algorithm

Minimal error boundary



Gradient Domain blending (1D)

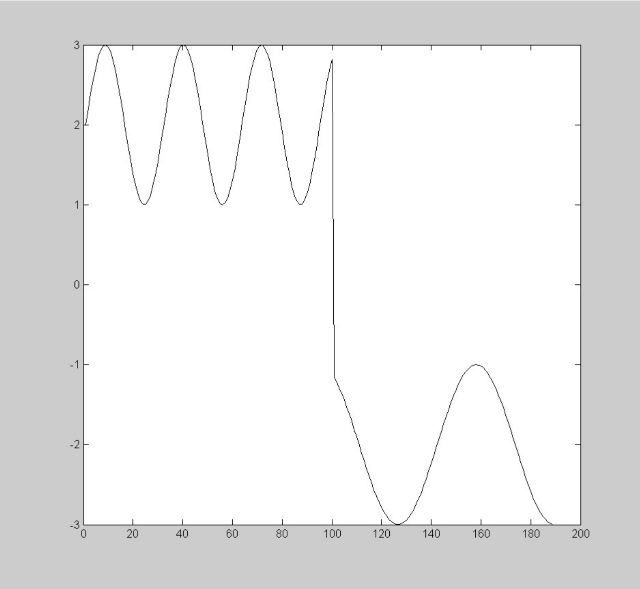
Two signals

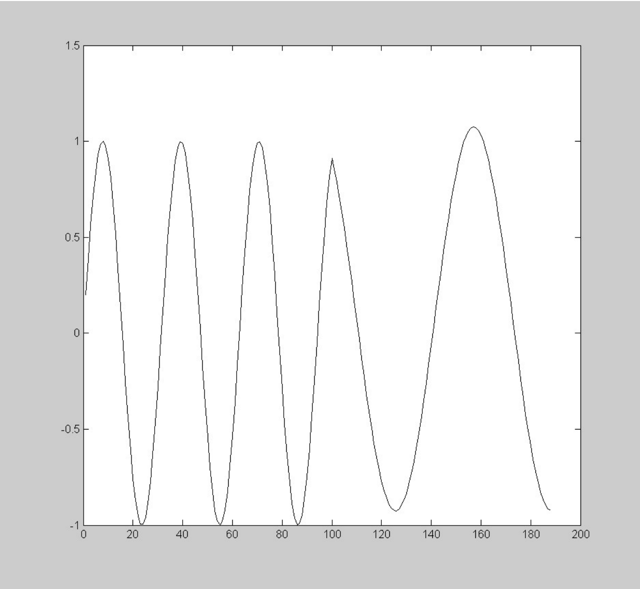


bright

dark

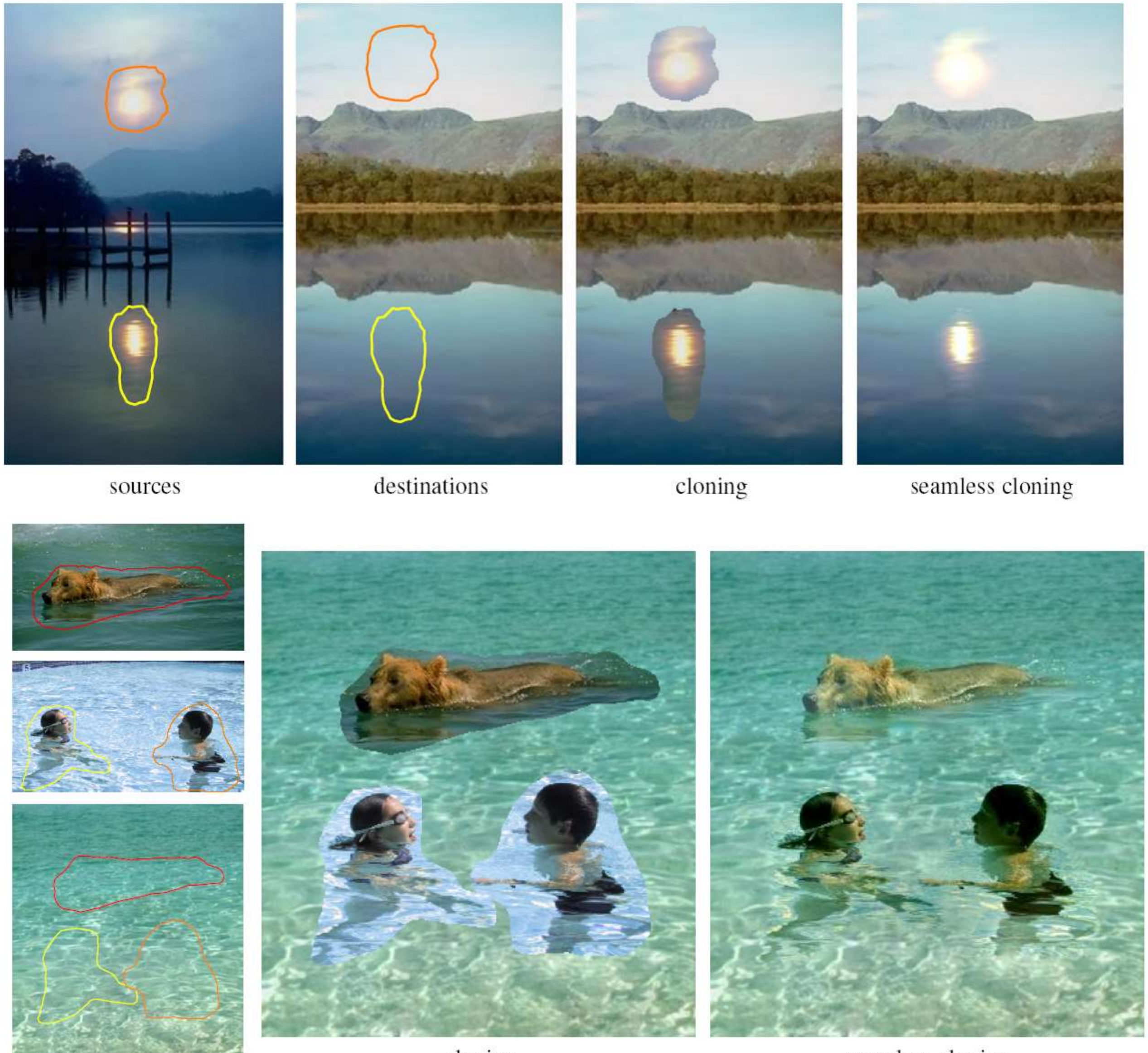
Regular blending





Blending derivatives

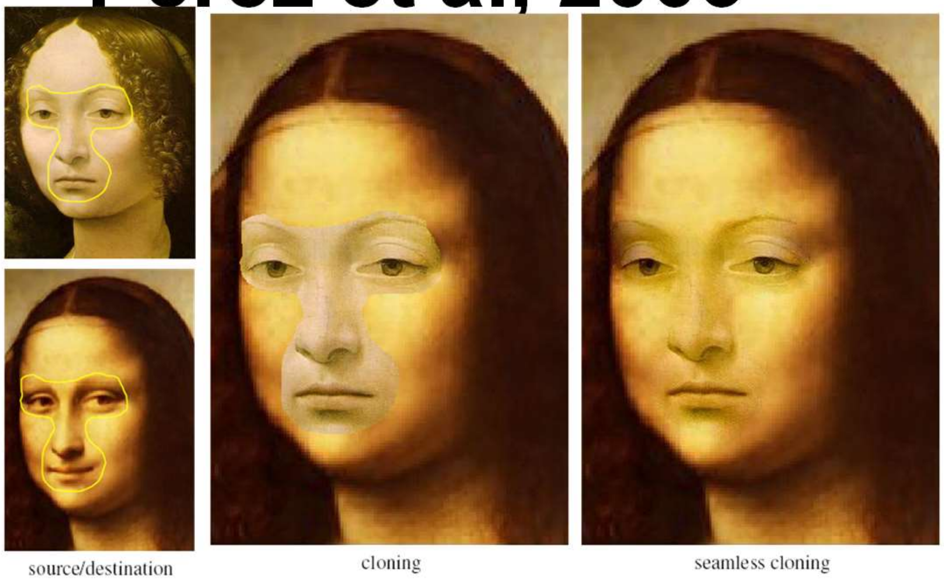
Perez et al.. 2003



sources destinations cloning seamless cloning

sources/destinations cloning seamless cloning

Perez et al, 2003



editing

Limitations:

- Can't do contrast reversal (gray on black -> gray on white)
- Colored backgrounds "bleed through"
- Images need to be very well aligned