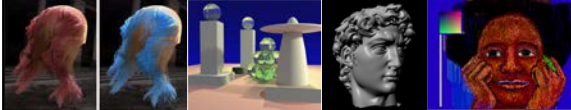


Advanced Computer Graphics

CSE 190 [Spring 2015], Lecture 5

Ravi Ramamoorthi

<http://www.cs.ucsd.edu/~ravir>



To Do

- Assignment 1, Due Apr 24.
 - This lecture only extra credit and clear up difficulties
- Questions/difficulties so far in doing assignment?

Digital Image Compositing

1996: Academy scientific and engineering achievement award (oscar ceremony) "For their pioneering inventions in digital image compositing"



Smith Duff Catmull Porter

Image Compositing

Separate an image into elements

- Each part is rendered separately
- Then pasted together or composited into a scene



Many slides courtesy Tom Funkhouser

Outline

Compositing

- Blue screen matting
- Alpha channel
- Porter-Duff compositing algebra (Siggraph 84)

Morphing (Beier-Neely, Siggraph 92)

Blue Screen Matting

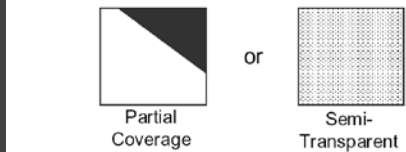
- Photograph or create image of object against blue screen (blue usually diff from colors like skin)
- Then extract foreground (non-blue pixels)
- Add (composite) to new image
- Problem: aliasing [hair] (no notion of partial coverage/blue)



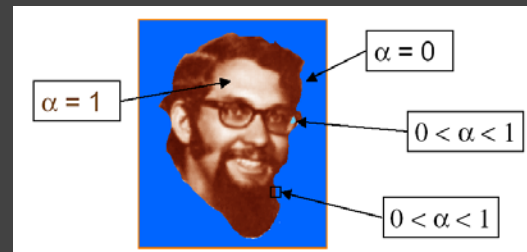
Alpha Channel

- In general, 32 bit RGBA images
- Alpha encodes coverage (0=transparent, 1=opaque)
- Simple compositing: $OUT = \alpha F + (1 - \alpha)B$

- Example: $\alpha = 0.3$



Alpha Channel



Pixels with Alpha: Conventions

Pre-multiplication

- Color $C = (r, g, b)$ and coverage alpha is often represented as $(\alpha r, \alpha g, \alpha b, \alpha)$
- One benefit: color components αF directly displayed (analogous to homogeneous coordinates)

What is (α, C) for the following?

- $(0, 1, 0, 1)$ = Full green, full coverage
- $(0, \frac{1}{2}, 0, 1)$ = Half green, full coverage
- $(0, \frac{1}{2}, 0, \frac{1}{2})$ = Full green, half (partial) coverage
- $(0, \frac{1}{2}, 0, 0)$ = No coverage

Compositing with Alpha

- Suppose we put A over B over background G



- How much of B is blocked by A?

$$\alpha_A$$

- How much of B shows through A?

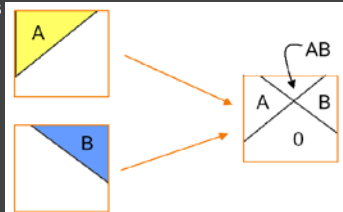
$$(1 - \alpha_A)$$

- How much of G shows through both A and B?

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

Opaque Objects

- In this case, α controls the amount of pixel covered (as in blue screening).
- How to combine 2 partially covered pixels? 4 possible outcomes



Outline

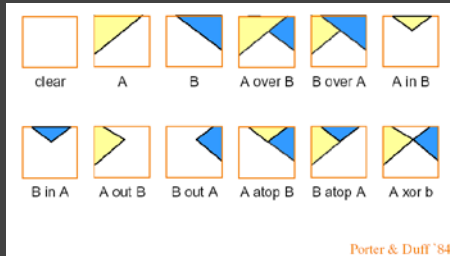
Compositing

- Blue screen matting
- Alpha channel
- Porter-Duff compositing algebra (Siggraph 84)

Morphing (Beier-Neely, Siggraph 92)

Compositing Algebra

- 12 reasonable combinations (operators)



Computing Colors with Compositing

- Coverages shown previously only examples
- We only have α , not exact coverage, so we assume coverages of A and B are uncorrelated
- How to compute net coverage for operators?

Example: $C = A \text{ over } B$

- For colors that are not premultiplied:

- $C = \alpha_A A + (1 - \alpha_A) \alpha_B B$
- $\alpha = \alpha_A + (1 - \alpha_A) \alpha_B$

- For colors that are premultiplied:

- $C' = A' + (1 - \alpha_A) B'$
- $\alpha = \alpha_A + (1 - \alpha_A) \alpha_B$



Assumption:
coverages of A and B
are uncorrelated
for each pixel

Image Compositing Example



Jurassic Park 93

Outline

Compositing

- Blue screen matting
- Alpha channel
- Porter-Duff compositing algebra (Siggraph 84)

Morphing (Beier-Neely, Siggraph 92)

Examples

- Famous example: Michael Jackson Black and White Video (Nov 14, 1991).
 - <https://www.youtube.com/watch?v=YpTNvUlwjFk>
- Easy enough to implement: assignment in many courses (we show example from CMU course):
 - No music, but the good poor man's alternative



Simple Cross-Dissolve

- Blend images with "over" operator
 - alpha of bottom image is 1.0
 - alpha of top image varies from 0.0 to 1.0

$$\text{blend}(i,j) = (1-t) \text{src}(i,j) + t \text{dst}(i,j) \quad (0 \leq t \leq 1)$$

src

t = 0.0

blend

t = 0.5

dst

t = 1.0

← t →

The idea in morphing

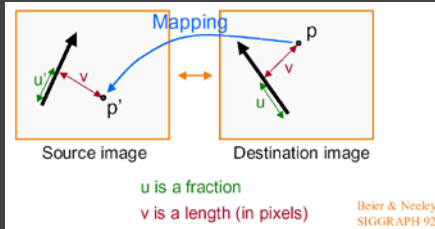
- User marks line segments
- These are used to warp image 1 and image 2
- Images are then blended
- Key step is warping
 - Why is it needed? Why not just cross-dissolve or blend two images based on alpha (how far between them)
 - How is it to be done with line segments?

Feature-Based Warping

- To warp image 1 into image 2, we must establish correspondence between like features
- Then, those features transform (and rest of image moves with them)
- In Beier-Neely, features are user-specified line segments (nose, face outline etc.)
- Warping is an image transformation (generally more complex than scale or shift, but same basic idea)
- Morphing takes two warped images, cross-dissolves

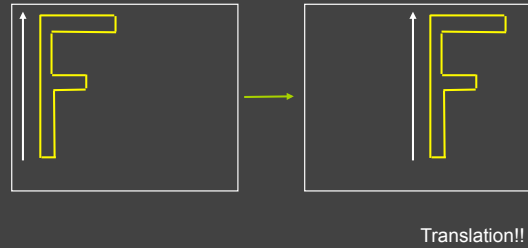
Warping with Line Segments

- We know how line warps, but what about whole img?
- Given p in dest image, where is p' in source image?



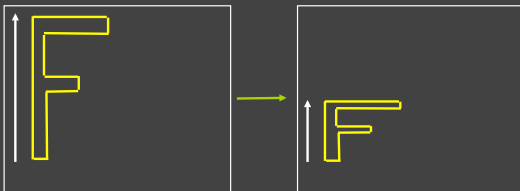
Warping with one Line Pair

- What happens to the F?



Warping with one Line Pair

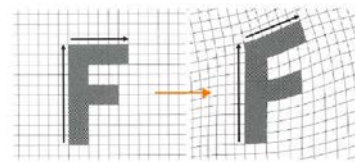
- What happens to the F?



Similar ideas apply to rotation,
other similarity transforms

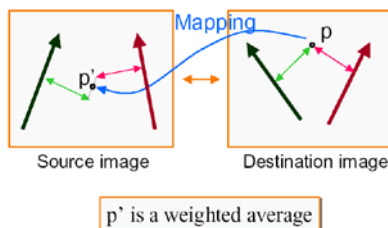
Warping with Multiple Line Pairs

- Use weighted combination of points defined by each pair of corresponding lines



Details

- Use weighted combination of points defined by each pair of corresponding lines



Weighting effect of each line pair

- To weight the contribution of each line pair, Beier & Neeley use:

$$weight[i] = \left(\frac{length[i]^p}{a + dist[i]} \right)^b$$

Where:

- $length[i]$ is the length of $L[i]$
- $dist[i]$ is the distance from X to $L[i]$
- a, b, p are constants that control the warp

Warping Pseudocode

```

WarpImage(Image, L[...], L[...])
begin
  foreach destination pixel p do
    psum = (0,0)
    wsum = 0
    foreach line L[i] in destination do
      p'[i] = p transformed by (L[i], L'[i])
      psum = psum + p'[i] * weight[i]
      wsum += weight[i]
    end
    p' = psum / wsum
    Result(p) = Image(p')
  end
end

```

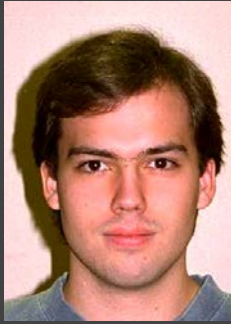
Morphing Pseudocode

```

GenerateAnimation(Image0, L0[...], Image1, L1[...])
begin
  foreach intermediate frame time t do
    for i = 1 to number of line pairs do
      L[i] = line t-th of the way from L0 [i] to L1 [i]
    end
    Warp0 = WarpImage(Image0, L0, L)
    Warp1 = WarpImage(Image1, L1, L)
    foreach pixel p in FinalImage do
      Result(p) = (1-t) Warp0 + t Warp1
    end
  end
end

```

Examples



Bonus: Reconstruction

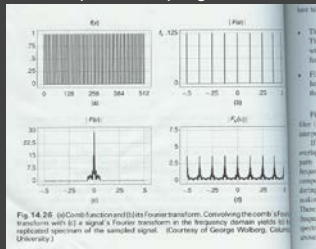
- Section 14.10.5 of textbook (in handout)
- Some interesting, more technical ideas
- Discuss briefly if time permits

Discrete Reconstruction

Equivalent to multiplying by comb function (a)

Convolution with similar fn in frequency domain (b). Separation in frequency domain depends on spatial sampling rate

Replicated Fourier spectra (when is this safe?)



Replicated Fourier Spectra

- One can window to eliminate unwanted spectra
- Equivalent to convolution with sinc
- No aliasing if spectra well enough separated (initial spatial sampling rate high enough)
- In practice, we use some reconstruction filter (not sinc), such as triangle or Mitchell filter

