Drag-and-Drop Pasting

SIGGRAPH 2006
Submission ID# 247
Image Blending + Image Carving

© Kenneth Kwan

Slides Modified from Alexei Efros, CMU,
Compositing Procedure

1. Extract Sprites (e.g. using *Intelligent Scissors* in Photoshop)

2. Blend them into the composite (in the right order)
Just replacing pixels rarely works

Problems: boundaries & transparency (shadows)
Two Problems:

Semi-transparent objects

Pixels too large
Solution: alpha channel

Add one more channel:
- Image(R,G,B,alpha)

Encodes transparency (or pixel coverage):
- Alpha = 1: opaque object (complete coverage)
- Alpha = 0: transparent object (no coverage)
- 0<Alpha<1: semi-transparent (partial coverage)

Example: alpha = 0.3
Alpha Blending

\[ l_{\text{comp}} = \alpha l_{\text{fg}} + (1-\alpha) l_{\text{bg}} \]
TEXAS WEATHER: DALLAS/FT. WORTH
30% AM SHOWERS, 66
Multiple Alpha Blending

So far we assumed that one image (background) is opaque.

If blending semi-transparent sprites (the “A over B” operation):

\[ I_{\text{comp}} = \alpha_a I_a + (1-\alpha_a)\alpha_b I_b \]

\[ \alpha_{\text{comp}} = \alpha_a + (1-\alpha_a)\alpha_b \]

Note: sometimes alpha is premultiplied:

\[ \text{im}(\alpha R, \alpha G, \alpha B, \alpha) : \]

\[ I_{\text{comp}} = I_a + (1-\alpha_a)I_b \]

(same for alpha!)
Flash Matting

Siggraph 2006
Submission ID #220
Alpha Hacking…

No physical interpretation, but it smoothes the seams
Feathering

Encoding as transparency

$$I_{\text{blend}} = \alpha I_{\text{left}} + (1-\alpha)I_{\text{right}}$$
Setting alpha: simple averaging

Alpha = .5 in overlap region
Setting alpha: center seam

\[ \text{Alpha} = \text{logical}(d\text{trans}1 > d\text{trans}2) \]
Setting alpha: blurred seam

Distance transform

Alpha = blurred
Setting alpha: center weighting

\[
\text{Alpha} = \frac{d\text{trans}1}{d\text{trans}1 + d\text{trans}2}
\]
Affect of Window Size
Affect of Window Size
Good Window Size

“Optimal” Window: smooth but not ghosted
What is the Optimal Window?

To avoid seams
  • window \( \geq \) size of largest prominent feature

To avoid ghosting
  • window \( \leq 2 \times \) size of smallest prominent feature
Gaussian pyramid is smooth => can be subsampled

Laplacian pyramid has narrow band of frequency => compressed
Pyramid Blending

Left pyramid

blend

Right pyramid
Pyramid Blending
layer level 4

layer level 2

layer level 0

left pyramid  right pyramid  blended pyramid
Laplacian Pyramid: Blending

General Approach:

1. Build Laplacian pyramids $LA$ and $LB$ from images $A$ and $B$
2. Build a Gaussian pyramid $GR$ from selected region $R$
3. Form a combined pyramid $LS$ from $LA$ and $LB$ using nodes of $GR$ as weights:
   - $LS(i,j) = GR(i,j) \times LA(i,j) + (1 - GR(i,j)) \times LB(i,j)$
4. Collapse the $LS$ pyramid to get the final blended image
Horror Photo

© prof. dmartin
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 mask
2-band Blending

Low frequency ($\lambda > 2$ pixels)

High frequency ($\lambda < 2$ pixels)
Linear Blending
2-band Blending
Gradient Domain

In Pyramid Blending, we decomposed our image into 2\textsuperscript{nd} derivatives (Laplacian) and a low-res image.

Let us now look at 1\textsuperscript{st} derivatives (gradients):

- No need for low-res image
  - captures everything (up to a constant)

- Idea:
  - Differentiate
  - Blend
  - Reintegrate
Gradient Domain blending (1D)

Two signals

Regular blending

Blending derivatives
Gradient Domain Blending (2D)

Trickier in 2D:

- Take partial derivatives $dx$ and $dy$ (the gradient field)
- Fiddle around with them (smooth, blend, feather, etc)
- Reintegrate
  - But now integral($dx$) might not equal integral($dy$)
- Find the most agreeable solution
  - Equivalent to solving Poisson equation
  - Can use FFT, deconvolution, multigrid solvers, etc.
Comparisons: Levin et al, 2004

Pyramid blending

Feathering

Pyramid blending on the gradients

GIST1
Limitations:

- Can’t do contrast reversal (gray on black -> gray on white)
- Colored backgrounds “bleed through”
- Images need to be very well aligned
Don’t blend, CUT!

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

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