Image Compositing
Compositing Procedure

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

2. Blend them into the composite (in the right order)
Need blending
Alpha Blending / Feathering

\[ I_{\text{blend}} = \alpha I_{\text{left}} + (1-\alpha)I_{\text{right}} \]
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

Alpha = blurred
Setting alpha: center weighting

\[ \text{Alpha} = \frac{\text{dtrans1}}{\text{dtrans1} + \text{dtrans2}} \]
Affect of Window Size

![Diagram showing the affect of window size with left and right markers.]
Affect of Window Size
Good Window Size

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

To avoid seams
• window = size of largest prominent feature

To avoid ghosting
• window <= 2*size of smallest prominent feature

Natural to cast this in the *Fourier domain*
• largest frequency <= 2*size of smallest frequency
• image frequency content should occupy one “octave” (power of two)
What if the Frequency Spread is Wide

Idea (Burt and Adelson)

- Compute $F_{\text{left}} = \text{FFT}(I_{\text{left}})$, $F_{\text{right}} = \text{FFT}(I_{\text{right}})$
- Decompose Fourier image into octaves (bands)
  - $F_{\text{left}} = F_{\text{left}}^{1} + F_{\text{left}}^{2} + \ldots$
- Feather corresponding octaves $F_{\text{left}}^{i}$ with $F_{\text{right}}^{i}$
  - Can compute inverse FFT and feather in spatial domain
- Sum feathered octave images in frequency domain

Better implemented in *spatial domain*
Octaves in the Spatial Domain

Lowpass Images

Bandpass Images
Pyramid Blending

Left pyramid

blend

Right pyramid
Pyramid Blending
laplacian level 4

laplacian level 2

laplacian 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:
   - $LS(i,j) = GR(l,j) \times LA(l,j) + (1-GR(l,j)) \times LB(l,j)$
4. Collapse the $LS$ pyramid to get the final blended image
Blending Regions
Horror Photo

© david dmartin (Boston College)
Results from this class (fall 2005)
Season Blending (St. Petersburg)
Season Blending (St. Petersburg)
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 ($\lambda > 2$ pixels)

High frequency ($\lambda < 2$ 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?
Segment the mosaic

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

overlapping blocks

vertical boundary

overlap error

min. error boundary
Seam Carving

Seam Carving for Content-Aware Image Resizing

Shai Avidan
Mitsubishi Electric Research Labs

Ariel Shamir
The Interdisciplinary Center & MERL
Graphcuts

What if we want similar “cut-where-things-agree” idea, but for closed regions?

• Dynamic programming can’t handle loops
Minimum cost cut can be computed in polynomial time
(max-flow/min-cut algorithms)
Actually, for this example, DP will work just as well…
Interactive segmentation using graphcuts
Gradient Domain

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

Let us now look at 1st 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

bright
dark
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 $\int dx$ might not equal $\int dy$
- Find the most agreeable solution
  - Equivalent to solving Poisson equation
  - Can use FFT, deconvolution, multigrid solvers, etc.
Perez et al., 2003
Limitations:

- Can’t do contrast reversal (gray on black -> gray on white)
- Colored backgrounds “bleed through”
- Images need to be very well aligned
Real-Time Gradient-Domain Painting

James McCann∗
Carnegie Mellon University

Nancy S. Pollard†
Carnegie Mellon University

Code available!

Our very own Jim McCann::

James McCann
Real-Time Gradient-Domain Painting, SIGGRAPH 2009
Putting it all together

Compositing images

• Have a clever blending function
  – Feathering
  – Center-weighted
  – blend different frequencies differently
  – Gradient based blending

• Choose the right pixels from each image
  – Dynamic programming – optimal seams
  – Graph-cuts

Now, let’s put it all together:

• Interactive Digital Photomontage, 2004 (video)
Interactive Digital Photomontage

Aseem Agarwala, Mira Dontcheva
Maneesh Agrawala, Steven Drucker, Alex Colburn
Brian Curless, David Salesin, Michael Cohen