Image Composition

© NASA

Modelled after lecture by Alexei Efros.
Slides by Efros, Durand, Freeman, Hays, Fergus, Lazebnik, Agarwala, Shanir, and Perez.

Image Compositing

Slide credit: A. Efros

Compositing Procedure

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

Slide credit: A. Efros

2. Blend them into the composite (in the right order)

Composite by David Dewey

Slide credit: A. Efros

Need blending

Slide credit: A. Efros

Alpha Blending / Feathering

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

Slide credit: A. Efros

Setting alpha: simple averaging

Alpha = .5 in overlap region

Slide credit: A. Efros
Setting alpha: center weighting

\[
\text{Distance transform}
\]

\[
\text{Ghost!}
\]

\[ \text{Alpha} = \frac{d\text{trans}_1}{(d\text{trans}_1+d\text{trans}_2)} \]  

Slide credit: A. Efros

Affect of Window Size

Slide credit: A. Efros

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)

Slide credit: A. Efros

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**  

Slide credit: A. Efros
Octaves in the Spatial Domain

Bandpass Images

Lowpass Images

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:
   - $LS(i,j) = GR(i,j)^*LA(i,j) + (1-GR(i,j))^*LB(i,j)$
4. Collapse the $LS$ pyramid to get the final blended image
Laplacian Pyramid Blending

Problems with blending

Misaligned (moving) objects become ghosts

Seams
Segment the images
- Single source image per segment
- Find optimal seams between segments
- Optionally blend across seams

Seams in texture synthesis
overlapping blocks vertical boundary

overlap error min. error boundary

Seam Carving

Seam Carving
Seam Carving

$E(I) = \frac{\partial I}{\partial x} + \frac{\partial I}{\partial y}$

$\Rightarrow s^* = \arg \min_s E(s)$

Seam Carving

Removal of vertical seams

Removal of horizontal seams

Seams with Graphcuts

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

- Dynamic programming can’t handle loops
Seams with Graph cuts

Minimum cost cut can be computed in polynomial time
(max-flow/min-cut algorithms)

Boykov&Jolly, ICCV’01

Seams with Graph Cuts

Lazy Snapping
Interactive segmentation using graphcuts

Problem with seams
What if colors/intensities are different?

Problem with seams
What if colors/intensities are different?

Gradient domain image editing
Motivation:
Human visual system is very sensitive to gradient
Gradient encode edges and local contrast quite well

Approach:
Edit in the gradient domain
Reconstruct image from gradient

Various instances of this idea, I’ll mostly follow Perez et al. Siggraph 2003
http://research.microsoft.com/vision/cambridge/papers/perez_sighgraph03.pdf
1D example: minimization

Minimize derivatives to interpolate

\[
\begin{align*}
\min (f_2-f_1)^2 & \\
\min (f_2-f_3)^2 & \quad \text{with} \quad l_1=6 \\
\min (f_3-f_4)^2 & \\
\min (f_4-f_6)^2 & \quad l_6=1 \\
\end{align*}
\]

1D example: derivatives

Minimize derivatives to interpolate

\[
\begin{align*}
\frac{df_2}{df_1} &= 2f_2 + 2f_3 - 2f_5 - 12 \\
\frac{df_3}{df_2} &= 2f_2 - 2f_3 + 2f_5 - 2f_4 \\
\frac{df_4}{df_3} &= 2f_3 - 2f_4 + 2f_5 - 2f_6 \\
\frac{df_5}{df_4} &= 2f_5 - 2f_6 + 2f_4 - 2f_2 \\
\end{align*}
\]

\[
\begin{pmatrix}
4 & -2 & 0 & 0 \\
-2 & 4 & -2 & 0 \\
0 & -2 & 4 & -2 \\
0 & 0 & -2 & 4 \\
\end{pmatrix}
\begin{pmatrix}
f_2 \\
f_3 \\
f_4 \\
f_5 \\
\end{pmatrix}
=
\begin{pmatrix}
12 \\
0 \\
0 \\
0 \\
\end{pmatrix}
\]

Pretty much says that second derivative should be zero

\((-1 2 -1)\) is a second derivative filter

1D example: set derivatives to zero

Minimize derivatives to interpolate

\[
\begin{align*}
\frac{df_2}{df_1} &= 2f_2 + 2f_3 - 2f_5 - 12 \\
\frac{df_3}{df_2} &= 2f_2 - 2f_3 + 2f_5 - 2f_4 \\
\frac{df_4}{df_3} &= 2f_3 - 2f_4 + 2f_5 - 2f_6 \\
\frac{df_5}{df_4} &= 2f_5 - 2f_6 + 2f_4 - 2f_2 \\
\end{align*}
\]

Membrane interpolation

Laplace equation (a.k.a. membrane equation)

\[
\min \frac{1}{|\Omega|} \int_{\Omega} |\nabla f|^2 \quad \text{with} \quad f_\partial = f^+ |\partial \Omega
\]
Membrane interpolation

Laplace equation (a.k.a. membrane equation)

\[ \min \int_{\Omega} |\nabla f|^2 \text{ with } f|_{\partial \Omega} = f^+|_{\partial \Omega} \]

Mathematicians will tell you there is an Associated Euler-Lagrange equation:

- Where the Laplacian \( \Delta \) is similar to -1 2 -1 in 1D

Kind of the idea that we want a minimum, so we kind of derive and get a simpler equation

Slide credit: F. Durand

Seamless Poisson cloning

Given vector field \( v \) (pasted gradient), find the value of \( f \) in unknown region that optimize:

Previously, \( v \) was null

\[ \min \int_{\Omega} |\nabla f - v|^2 \text{ with } f|_{\partial \Omega} = f^+|_{\partial \Omega} \]

Pasted gradient Mask

\( f \)

Unknown region

Slide credit: F. Durand

What if \( v \) is not null: 2D

Variational minimization (integral of a functional)

with boundary condition

\[ \min \int_{\Omega} |\nabla f - v|^2 \text{ with } f|_{\partial \Omega} = f^+|_{\partial \Omega} \]

Euler-Lagrange equation:

\[ \Delta f = \text{div} v \text{ over } \Omega, \text{ with } f|_{\partial \Omega} = f^+|_{\partial \Omega} \]

where \( \text{div} = \frac{\partial v}{\partial x} + \frac{\partial v}{\partial y} \) is the divergence of \( v = (u, v) \)

(Compared to Laplace, we have replaced \( \Delta = 0 \) by \( \Delta = \text{div} \))

Slide credit: F. Durand

Discrete 1D example: minimization

Copy to

Min ((f_2-f_1)-1)^2
Min ((f_3-f_2)-(-1))^2
Min ((f_4-f_3)-2)^2
Min ((f_5-f_4)-(-1))^2
Min ((f_6-f_5)-(-1))^2

\( f_1 = 6 \)
\( f_2 = 1 \)

Denote it \( Q \)

Slide credit: F. Durand

1D example: minimization

Copy to

Min ((f_2-6)-1)^2
Min ((f_3-f_2)-(-1))^2
Min ((f_4-f_3)-2)^2
Min ((f_5-f_4)-(-1))^2
Min ((f_6-f_5)-(-1))^2

\( f_2 = 49-14f_1 \)
\( f_3 = f_2^2 + 1-2f_2f_1 + 2f_2^2 - 2f_1 \)
\( f_4 = f_3^2 + 4+2f_2f_3 \)
\( 4f_4 + 4f_3 \)
\( f_5 = f_4^2 + 1-2f_4f_2 + 2f_4^2 - 2f_2 \)
\( f_6 = f_5^2 + 4-4f_4 \)

Slide credit: F. Durand

1D example: big quadratic

Copy to

Min (f_2^2 + 1-2f_2f_1 + 2f_2^2 - 2f_1)
\( + f_3^2 + f_2^2 + 1-2f_2f_3 + 2f_3 - 2f_2 \)
\( + f_4^2 + f_3^2 + 4-2f_3f_2 + 4f_3 + 4f_2 \)
\( + f_5^2 + f_4^2 + 1-2f_4f_3 + 2f_4^2 - 2f_3 \)
\( + f_6^2 + f_5^2 + 4-4f_4 \)

Denote it \( Q \)

Slide credit: F. Durand
1D example: derivatives

\[
\begin{align*}
\frac{d^2Q}{d^2x} &= f_2 + f_2 - 2f_3 - 16 \\
\frac{d^2Q}{d^2y} &= 2f_3 - f_3 - 2 + f_3 - 4 + 4f_4 - 4f_2 \\
\frac{d^2Q}{d^2z} &= 2f_4 - f_4 - 4 + 2f_4 - f_4 - 2 \\
\frac{d^2Q}{d^2f} &= 2f_5 - f_5 + 2 + 2f_5 - 4
\end{align*}
\]

Denote it \( Q \)

Slide credit: F. Durand

1D example: set derivatives to zero

\[
\begin{align*}
\frac{d^2Q}{d^2x} &= 2f_2 + f_2 - 2f_3 - 16 \\
\frac{d^2Q}{d^2y} &= 2f_3 - f_3 - 2 + f_3 - 4 + 4f_4 - 4f_2 \\
\frac{d^2Q}{d^2z} &= 2f_4 - f_4 - 4 + 2f_4 - f_4 - 2 \\
\frac{d^2Q}{d^2f} &= 2f_5 - f_5 + 2 + 2f_5 - 4
\end{align*}
\]

\[\Rightarrow\]

Slide credit: F. Durand

1D example: remarks

Matrix is sparse
Matrix is symmetric
Everything is a multiple of 2
- because square and derivative of square
Matrix is a convolution (kernel -2 4 -2)
Matrix is independent of gradient field. Only RHS is
Matrix is a second derivative

Slide credit: F. Durand

What if \( v \) is not null: 2D

Variational minimization (integral of a functional) with boundary condition

\[
\min \int_{\Omega} |\nabla f - v|^2 \text{ with } f|_{\partial\Omega} = f'|_{\partial\Omega}.
\]

Euler-Lagrange equation:

\[
\Delta f = \text{div} v \text{ over } \Omega, \text{ with } f|_{\partial\Omega} = f'|_{\partial\Omega}
\]

where \( \text{div} v = \frac{\partial v}{\partial x} + \frac{\partial v}{\partial y} \) is the divergence of \( v = (u,v) \)

(Compared to Laplace, we have replaced \( \Delta = 0 \) by \( \Delta = \text{div} \))

Slide credit: F. Durand

Discrete Poisson solver

Two approaches:
- \( \text{Minimize variational problem} \)
- \( \text{Solve Euler-Lagrange equation} \)

In practice, variational is best

\( \text{Finite differences over 4 pixel neighbors} \)
\( \text{Partial derivatives are easy on pairs} \)
\( \text{Same for the discretization of} v \)

In both cases, need to discretize derivatives

Slide credit: F. Durand
Discrete Poisson solver

Minimize variational problem
\[ \min \int_{\Omega} \|\nabla f - v\|^2 \text{ with } f|_{\partial\Omega} = f^|_{\partial\Omega} \]

Discretized

\[ \min \sum_{p \in \Omega} (f_p - f_q - v_{pq})^2, \text{ with } f_p = f^p_p \text{ for all } p \in \partial\Omega \]

Discretized gradient

\[ v : g(p) - g(q) \]

Boundary condition

Rearrange and call \( N_p \) the neighbors of \( p \)

for all \( p \in \Omega \),

\[ \sum_{q \in N_p} f_q = \sum_{q \in N_p} f^p_q + \sum_{q \in N_p} v_{pq} \]

Only for boundary pixels

Big yet sparse linear system

Slide credit: F. Durand

Image Composition Results

Figure 2: Concealment. By importing seamlessly a piece of the background, complete objects, parts of objects, and undesirable artifacts can easily be hidden. In both examples, multiple strokes (not shown) were used.

Perez et al. SIGGRAPH 03

Figure 5: Monochrome transfer. In some cases, such as texture transfer, the part of the source color remaining after seamless cloning might be undesirable. This is fixed by turning the source image monochrome beforehand.

Perez et al. SIGGRAPH 03
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)
Scene Completion Using Millions of Photographs

James Hays and Alexei A. Efros
SIGGRAPH 2007

Slides by J. Hays and A. Efros

Efros and Leung result

Scene Matching for Image Completion
Data

2.3 Million unique images from Flickr groups and keyword searches.

Scene Completion Result

The Algorithm

Input image  Scene Descriptor  Image Collection

20 completions  Context matching + blending  200 matches

Scene Matching
Scene Descriptor

Context Matching
Result Ranking

We assign each of the 200 results a score which is the sum of:

- The scene matching distance
- The context matching distance (color + texture)
- The graph cut cost
Why does it work?

10 nearest neighbors from a collection of 20,000 images

10 nearest neighbors from a collection of 2 million images