Image Warping and Morphing

Lecture 7

Admin

• Here is feedback for Assignments 1 & 2
• Will post solutions after this class
• Assignment 3 out tomorrow – will email list
• Projects: If you haven’t come to see me about your project, please do so!

Best blending examples

• Paul Gentry

Best blending examples

• Marco

Best blending examples

• Roger Xue

Best blending examples

• Alfredo
Best blending examples

- Merve

Admin

- Aloysha Efros (CMU) talk on Wednesday 11am
  - Next week's lecture will cover some of his work

  • Title: Ask not “what is this?” but rather “what is this like?”

Outline

- Photo Tourism
- Seam Carving
- Morphing

Photo Tourism: Exploring Photo Collections in 3D

Noah Snavely
Steven M. Seitz
University of Washington
Richard Szeliski
Microsoft Research

© 2006 Noah Snavely
Photo Tourism

Related work
- Image-based modeling
  - Debevec, et al. SIGGRAPH 1996
  - Schaffalitzky and Zisserman ECCV 2002
  - Brown and Lowe 3DIM 2005
- Image-based rendering
  - Photorealistic IBR:
    - Levoy and Hanrahan, SIGGRAPH 1996
    - Setlitz and Dyer, SIGGRAPH 1996
    - and many others

Feature detection
- Detect features using SIFT [Lowe, IJCV 2004]
**Pairwise feature matching**

- Match features between each pair of images

**Feature detection**

Detect features using SIFT [Lowe, IJCV 2004]

**Feature matching**

Match features between each pair of images

**Refine matching using RANSAC [Fischler & Bolles 1987]**

to estimate fundamental matrices between pairs
Feature matching

Correspondence estimation
- Link up pairwise matches to form connected components of matches across several images.

Structure from motion
\[
\minimize f(R, T, P)
\]
- \(R\) = rotations
- \(T\) = positions
- \(P\) = 3D point locations

Incremental structure from motion

Photo Tourism overview
- Navigation
- Rendering
- Annotations
Object-based browsing

- Visibility
- Resolution
- Head-on view

Relation-based browsing

- Image A to the right of Image B
- Image A to the left of Image C
- Image A is detail of Image B
- Image A is zoom-out of Image D
- Image B to the right of Image C
- Image B to the left of Image D
- Image B is detail of Image C
- Image B is zoom-out of Image D
- Image C to the right of Image D
- Image C to the left of Image D
- Image C is detail of Image D
- Image C is zoom-out of Image D

Photo Tourism overview

- Scene reconstruction
- Input photographs

Rendering

- Navigation
- Rendering
- Annotations
Live Demo

Outline
- Photo Tourism
- Seam Carving
- Morphing

Seam Carving for Content-Aware Image Resizing
Shadi Avizieni
Ariel Shamir
Mitsubishi Electric Research Labs
The Interdisciplinary Center & MERL

- How to reduce width of this image?

Rescale it?

Crop it?

Seam removal
Seam (vertical) - Definition

\[ s^k = \{ (i', j') \}^{k=0}_{k=M} \text{ such that: } \forall i, |x(i') - x(i-1)| \leq k \]

- All the pixels in the seam are removed (shift row/column).
- Visual artifacts are visible only along the seam.

Search of optimal seam (k=1)

Solved through dynamic programming:

Scan every row in the image from j=2 to j=M-1 updating with local best choice.

\[ M_{ij} = \text{cumulative minimum energy at position } (i, j) \]

\[ M_{ij} = e(i, j) + \min\{ M_{i+1, j}, M_{i, j+1}, M_{i, j-1} \} \]

In the last row we pick the smallest entry and we backtrack a path choosing always the local minima.

Slide credit: Andrea Tagliasacchi.
Seam insertion

Object Removal

What energy function?

\[ r_1(l) = \frac{1}{2} \frac{\partial^2}{\partial x^2} + \frac{1}{2} \frac{\partial^2}{\partial y^2} \]

\[ r_{out}(l) = \frac{\|v*l + b\|_1}{\text{mean}(v*x-w(x,y)))} \]
Retargeting using face detector

Outline

- Photo Tourism
- Seam Carving
- Morphing

Morphing = Object Averaging

The aim is to find "an average" between two objects

- Not an average of two images of objects...
- ...but an image of the average object!
- How can we make a smooth transition in time?
  - Do a "weighted average" over time t

How do we know what the average object looks like?

- We haven't a clue!
- But we can often fake something reasonable
  - Usually required user/artist input

Averaging Points

What's the average of P and Q?

Linear interpolation
(Affine Combination):
New point aP + bQ,
defined only when a + b = 1
So aP + bQ = aP + (1-a)Q

P and Q can be anything:
- points on a plane (2D) or in space (3D)
- Colors in RGB or HSV (3D)
- Whole images (m-by-n D) etc.

Idea #1: Cross-Dissolve

Interpolate whole images:
Image_{halfway} = (1-t)*image_1 + t*image_2

This is called cross-dissolve in film industry

But what is the images are not aligned?

Slide credit: Alyosha Efros
Idea #2: Align, then cross-dissolve

Align first, then cross-dissolve
• Alignment using global warp – picture still valid

Slide credit: Alyosha Efros

Dog Averaging

What to do?
• Cross-dissolve doesn’t work
• Global alignment doesn’t work
  – Cannot be done with a global transformation (e.g., affine)
• Any ideas?
Feature matching!
• Nose to nose, tail to tail, etc.
• This is a local (non-parametric) warp

Slide credit: Alyosha Efros

Idea #3: Local warp, then cross-dissolve

Morphing procedure:
for every t,
1. Find the average shape (the “mean dog”)
  • local warping
2. Find the average color
  • Cross-dissolve the warped images

Slide credit: Alyosha Efros

Local (non-parametric) Image Warping

Need to specify a more detailed warp function
• Global warps were functions of a few (2, 4, 8) parameters
• Non-parametric warps u(x, y) and v(x, y) can be defined independently for every single location x, y!
• Once we know vector field u, v we can easily warp each pixel (use backward warping with interpolation)

Slide credit: Alyosha Efros

Image Warping – non-parametric

Move control points to specify a spline warp
Spline produces a smooth vector field

Slide credit: Alyosha Efros

Warp specification - dense

How can we specify the warp?
Specify corresponding spline control points
• interpolate to a complete warping function

But we want to specify only a few points, not a grid

Slide credit: Alyosha Efros
**Warp specification - sparse**

How can we specify the warp?
- Specify corresponding points
- interpolate to a complete warping function
- How do we do it?

How do we go from feature points to pixels?

Slide credit: Alyosha Efros

---

**Triangular Mesh**

1. Input correspondences at key feature points
2. Define a triangular mesh over the points
   - Same mesh in both images!
   - Now we have triangle-to-triangle correspondences
3. Warp each triangle separately from source to destination
   - How do we warp a triangle?
   - 3 points = affine warp!
   - Just like texture mapping

Slide credit: Alyosha Efros

---

**Detour: how to warp triangles**

Example: warping triangles

Given two triangles: ABC and A’B’C’ in 2D (12 numbers)
Need to find transform T to transfer all pixels from one to the other.
What kind of transformation is T?
How can we compute the transformation matrix:

\[
\begin{align*}
    \begin{bmatrix}
        x' \\
        y' \\
        1
    \end{bmatrix} &=
    \begin{bmatrix}
        a & b & c \\
        d & e & f \\
        0 & 0 & 1
    \end{bmatrix}
    \begin{bmatrix}
        x \\
        y \\
        1
    \end{bmatrix}
\end{align*}
\]

Slide credit: Alyosha Efros

---

**Image warping**

Given a coordinate transform \((x',y') = T(x,y)\) and a source image \(f(x,y)\), how do we compute a transformed image \(g(x',y') = f(T(x,y))\)?

Slide credit: Alyosha Efros

---

**Forward warping**

Send each pixel \(f(x,y)\) to its corresponding location \((x',y') = T(x,y)\) in the second image

Q: what if pixel lands “between” two pixels?

Slide credit: Alyosha Efros
Forward warping

Send each pixel \( f(x,y) \) to its corresponding location \( (x',y') = T(x,y) \) in the second image

Q: what if pixel lands "between" two pixels?
A: distribute color among neighboring pixels \( (x',y') \)
   - Known as "splatting"
   - Check out `griddata` in Matlab

Slide credit: Alyosha Efros

Inverse warping

Get each pixel \( g(x',y') \) from its corresponding location \( (x,y) = T^{-1}(x',y') \) in the first image

Q: what if pixel comes from "between" two pixels?
A: Interpolate color value from neighbors
   - nearest neighbor, bilinear, Gaussian, bicubic
   - Check out `interp2` in Matlab

Slide credit: Alyosha Efros

End of detour

Triangulations

A triangulation of set of points in the plane is a partition of the convex hull to triangles whose vertices are the points, and do not contain other points.

There are an exponential number of triangulations of a point set.
An \( O(n^3) \) Triangulation Algorithm

Repeat until impossible:
- Select two sites.
- If the edge connecting them does not intersect previous edges, keep it.

"Quality" Triangulations

Let \( \alpha(T) = (\alpha_1, \alpha_2, ..., \alpha_{3t}) \) be the vector of angles in the triangulation \( T \) in increasing order.
A triangulation \( T_1 \) will be "better" than \( T_2 \) if \( \alpha(T_1) > \alpha(T_2) \) lexicographically.
The Delaunay triangulation is the "best"
- Maximizes smallest angles

Improving a Triangulation

In any convex quadrangle, an edge flip is possible. If this flip improves the triangulation locally, it also improves the global triangulation.

If an edge flip improves the triangulation, the first edge is called illegal.

Illegal Edges

Lemma: An edge \( pq \) is illegal iff one of its opposite vertices is inside the circle defined by the other three vertices.
Proof: By Thales' theorem.

Theorem: A Delaunay triangulation does not contain illegal edges.
Corollary: A triangle is Delaunay iff the circle through its vertices is empty of other sites.
Corollary: The Delaunay triangulation is not unique if more than three sites are co-circular.

Naïve Delaunay Algorithm

Start with an arbitrary triangulation. Flip any illegal edge until no more exist.
Could take a long time to terminate.

Delaunay Triangulation by Duality

General position assumption: There are no four co-circular points.
Draw the dual to the Voronoi diagram by connecting each two neighboring sites in the Voronoi diagram.
Corollary: The DT may be constructed in \( O(n \log n) \) time.
This is what Matlab's \texttt{delaunay} function uses.
Image Morphing

We know how to warp one image into the other, but how do we create a morphing sequence?
1. Create an intermediate shape (by interpolation)
2. Warp both images towards it
3. Cross-dissolve the colors in the newly warped images

Warp interpolation

How do we create an intermediate warp at time $t$?
- Assume $t \in [0,1]$
- Simple linear interpolation of each feature pair
  \[(1-t)p_1 + tp_0\text{ for corresponding features } p_0 \text{ and } p_1\]

Morphing & matting

Extract foreground first to avoid artifacts in the background

Intermission Video

- You Tube