Image warping and stitching
May 5\textsuperscript{th}, 2015

Yong Jae Lee
UC Davis
Announcements

• PS2 due next Friday
Last time

• Interactive segmentation
• Feature-based alignment
  – 2D transformations
  – Affine fit
  – RANSAC
Alignment problem

- In alignment, we will fit the parameters of some transformation according to a set of matching feature pairs (“correspondences”).
Main questions

**Warping:** Given a source image and a transformation, what does the transformed output look like?

**Alignment:** Given two images, what is the transformation between them?

Slide credit: Kristen Grauman
Motivation for feature-based alignment: Recognition
Motivation for feature-based alignment: Medical image registration
Motivation for feature-based alignment: Image mosaics
Parametric (global) warping

Examples of parametric warps:

- Translation
- Rotation
- Aspect
- Affine
- Perspective

Source: Alyosha Efros
Parametric (global) warping

Transformation $T$ is a coordinate-changing machine:

$$p' = T(p)$$

What does it mean that $T$ is **global**?

- Is the same for any point $p$
- Can be described by just a few numbers (parameters)

Let’s represent $T$ as a matrix:

$$p' = Mp$$

$$\begin{bmatrix}
    x' \\
    y'
\end{bmatrix} = \begin{bmatrix}
    x \\
    y
\end{bmatrix}$$

Source: Alyosha Efros
Homogeneous coordinates

To convert to homogeneous coordinates:

\[(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}\]

homogeneous image coordinates

Converting *from* homogeneous coordinates

\[
\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w)
\]
Fitting an affine transformation

• Assuming we know the correspondences, how do we get the transformation?

\[
\begin{bmatrix}
    x'_i \\
    y'_i
\end{bmatrix}
= \begin{bmatrix}
    m_1 & m_2 \\
    m_3 & m_4
\end{bmatrix}
\begin{bmatrix}
    x_i \\
    y_i
\end{bmatrix}
+ \begin{bmatrix}
    t_1 \\
    t_2
\end{bmatrix}
\]
2D Affine Transformations

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

Affine transformations are combinations of …

- Linear transformations, and
- Translations

Parallel lines remain parallel
Projective Transformations

\[
\begin{bmatrix}
    x' \\
    y' \\
    w'
\end{bmatrix} = \begin{bmatrix}
    a & b & c \\
    d & e & f \\
    g & h & i
\end{bmatrix} \begin{bmatrix}
    x \\
    y \\
    w
\end{bmatrix}
\]

Projective transformations:
- Affine transformations, and
- Projective warps

Parallel lines do not necessarily remain parallel
Fitting an affine transformation

- Assuming we know the correspondences, how do we get the transformation?

\[
\begin{bmatrix}
  x'_i \\
  y'_i
\end{bmatrix} = \begin{bmatrix}
  m_1 & m_2 \\
  m_3 & m_4
\end{bmatrix} \begin{bmatrix}
  x_i \\
  y_i
\end{bmatrix} + \begin{bmatrix}
  t_1 \\
  t_2
\end{bmatrix}
\]

Slide credit: Kristen Grauman
RANSAC: General form

- **RANSAC loop:**
  1. Randomly select a seed group of points on which to base transformation estimate (e.g., a group of matches)
  2. Compute transformation from seed group
  3. Find *inliers* to this transformation
  4. If the number of inliers is sufficiently large, re-compute estimate of transformation on all of the inliers

- Keep the transformation with the largest number of inliers
RANSAC example: Translation

Putative matches
RANSAC example: Translation

Select *one* match, count *inliers*
RANSAC example: Translation

Select one match, count inliers
RANSAC example: Translation

Find “average” translation vector
RANSAC pros and cons

• Pros
  • Simple and general
  • Applicable to many different problems
  • Often works well in practice

• Cons
  • Lots of parameters to tune
  • Doesn’t work well for low inlier ratios (too many iterations, or can fail completely)
  • Can’t always get a good initialization of the model based on the minimum number of samples
Today

• Image mosaics
  – Fitting a 2D transformation
    • Homography
  – 2D image warping
  – Computing an image mosaic
HP frames commercials

- [http://www.youtube.com/watch?v=2RPl5vPEoQk](http://www.youtube.com/watch?v=2RPl5vPEoQk)
Mosaics

Obtain a wider angle view by combining multiple images.

Slide credit: Kristen Grauman
Panoramic Photos are old

- Sydney, 1875

Beirut, late 1800’s

Slide credit: James Hays
How to stitch together a panorama (a.k.a. mosaic)?

- **Basic Procedure**
  - Take a sequence of images from the same position
    - Rotate the camera about its optical center
  - Compute transformation between second image and first
  - Transform the second image to overlap with the first
  - Blend the two together to create a mosaic
  - (If there are more images, repeat)

- **...but wait**, why should this work at all?
  - What about the 3D geometry of the scene?
  - Why aren’t we using it?
Pinhole camera

- Pinhole camera is a simple model to approximate imaging process, perspective \textit{projection}.

If we treat pinhole as a point, only one ray from any given point can enter the camera.

Slide credit: Kristen Grauman
Fig from Forsyth and Ponce
Mosaics: generating synthetic views

Can generate any synthetic camera view as long as it has **the same center of projection**!
Obtain a wider angle view by combining multiple images.
Image reprojection

The mosaic has a natural interpretation in 3D

- The images are reprojected onto a common plane
- The mosaic is formed on this plane
- Mosaic is a synthetic wide-angle camera

Source: Steve Seitz
Image reprojection

Basic question
- How to relate two images from the same camera center?
  - how to map a pixel from PP1 to PP2

Answer
- Cast a ray through each pixel in PP1
- Draw the pixel where that ray intersects PP2

Observation:
Rather than thinking of this as a 3D reprojection, think of it as a 2D image warp from one image to another.

Source: Alyosha Efros
Image reprojection: Homography

A projective transform is a mapping between any two PPs with the same center of projection

- rectangle should map to arbitrary quadrilateral
- parallel lines aren’t preserved
- but must preserve straight lines

called **Homography**

\[
\begin{bmatrix}
w x' \\
w y' \\
w \\
p'
\end{bmatrix} = \begin{bmatrix}
* & * & * \\
* & * & * \\
* & * & * \\
1 & 1 & 1
\end{bmatrix} \begin{bmatrix}
x \\
y \\
l
\end{bmatrix}
\]

Source: Alyosha Efros
The projective plane

Why do we need homogeneous coordinates?
- represent points at infinity, homographies, perspective projection, multi-view relationships

What is the geometric intuition?
- a point in the image is a *ray* in projective space

- Each *point* \((x, y)\) on the plane is represented by a *ray* \((sx, sy, s)\)
  - all points on the ray are equivalent: \((x, y, 1) \cong (sx, sy, s)\)
To compute the homography given pairs of corresponding points in the images, we need to set up an equation where the parameters of $H$ are the unknowns...
Solving for homographies

\[
p' = Hp
\]

\[
\begin{bmatrix}
w x' \\
w y' \\
w
\end{bmatrix} = \begin{bmatrix}
h_{00} & h_{01} & h_{02} \\
h_{10} & h_{11} & h_{12} \\
h_{20} & h_{21} & h_{22}
\end{bmatrix} \begin{bmatrix}
x \\
y \\
1
\end{bmatrix}
\]

Upto a scale factor.
Constraint Frobenius norm of H to be 1.

Problem to be solved:

\[
\min \| Ah - b \|^2
\]

s.t. \( \| h \|^2 = 1 \)

where vector of unknowns \( h = [h_{00}, h_{01}, h_{02}, h_{10}, h_{11}, h_{12}, h_{20}, h_{21}, h_{22}]^T \)

Adapted from Devi Parikh
Solving for homographies

\[
\begin{bmatrix}
wx_i' \\
y_i' \\
w
\end{bmatrix} = \begin{bmatrix} h_{00} & h_{01} & h_{02} \\
h_{10} & h_{11} & h_{12} \\
h_{20} & h_{21} & h_{22}
\end{bmatrix} \begin{bmatrix} x_i \\
y_i \\
1
\end{bmatrix}
\]

\[wx_i' = h_{00}x_i + h_{01}y_i + h_{02}\]

\[wy_i' = h_{10}x_i + h_{11}y_i + h_{12}\]

\[w = h_{20}x_i + h_{21}y_i + h_{22}\]

\[x_i'(h_{20}x_i + h_{21}y_i + h_{22}) = h_{00}x_i + h_{01}y_i + h_{02}\]

\[y_i'(h_{20}x_i + h_{21}y_i + h_{22}) = h_{10}x_i + h_{11}y_i + h_{12}\]
Solving for homographies

\[
\begin{bmatrix}
x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1 & x_1 & -x'_1 \\
0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1 & x_1 & -y'_1 \\
x_n & y_n & 1 & 0 & 0 & 0 & -x'_n & x_n & -x'_n \\
0 & 0 & 0 & x_n & y_n & 1 & -y'_n & x_n & -y'_n \\
\end{bmatrix}
\begin{bmatrix}
h_{00} \\
h_{01} \\
h_{02} \\
h_{10} \\
h_{11} \\
h_{12} \\
h_{20} \\
h_{21} \\
h_{22} \\
\end{bmatrix}
= \begin{bmatrix}
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
\end{bmatrix}
\]

A \quad 2n \times 9 
\quad h \quad 9 
\quad 0 \quad 2n

Defines a least squares problem:

\[
\text{minimize } \|Ah - 0\|^2
\]

- Since \( h \) is only defined up to scale, solve for unit vector \( \hat{h} \) (i.e., \( \|h\|^2 = 1 \))
- Solution: \( \hat{h} = \text{eigenvector of } A^TA \text{ with smallest eigenvalue} \)
- Works with 4 or more points
To apply a given homography $H$

- Compute $p' = Hp$ (regular matrix multiply)
- Convert $p'$ from homogeneous to image coordinates

$$\begin{bmatrix}
wx' \\
w' \\
wy' \\
w
\end{bmatrix} = \begin{bmatrix}
* \\
* \\
* \\
1
\end{bmatrix} \begin{bmatrix}
x \\
y \\
1
\end{bmatrix} = \begin{bmatrix}
wx' \\
w' \\
w' \\
w
\end{bmatrix}$$
Today

• RANSAC for robust fitting
  – Lines, translation

• Image mosaics
  – Fitting a 2D transformation
    • Homography
  – 2D image warping
  – Computing an image mosaic
Image warping

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

>> help interp2
Bilinear interpolation

Sampling at $f(x,y)$:

$$f(x, y) = (1 - a)(1 - b) \cdot f[i, j] + a(1 - b) \cdot f[i + 1, j] + ab \cdot f[i + 1, j + 1] + (1 - a)b \cdot f[i, j + 1].$$
Recap: How to stitch together a panorama (a.k.a. mosaic)?

• Basic Procedure
  – Take a sequence of images from the same position
    • Rotate the camera about its optical center
  – Compute transformation (homography) between second image and first using corresponding points.
  – Transform the second image to overlap with the first.
  – Blend the two together to create a mosaic.
  – (If there are more images, repeat)
Image warping with homographies

image plane in front

black area where no pixel maps to

Source: Steve Seitz
Image rectification

Slide credit: Kristen Grauman
Analysing patterns and shapes

What is the shape of the b/w floor pattern?

The floor (enlarged)

Automatically rectified floor

Slide from Antonio Criminisi
Analysing patterns and shapes

From Martin Kemp *The Science of Art* (manual reconstruction)

Slide from Antonio Criminisi
Analysing patterns and shapes

What is the (complicated) shape of the floor pattern?

*St. Lucy Altarpiece, D. Veneziano*

Slide from Criminisi
Analysing patterns and shapes

From Martin Kemp, *The Science of Art (manual reconstruction)*
Analyzing patterns and shapes

The Ambassadors by Hans Holbein the Younger, 1533
Julian Beever: Manual Homographies

http://users.skynet.be/J.Beever/pave.htm
Changing camera center

Does it still work?

Source: Alyosha Efros
Recall: same camera center

Can generate synthetic camera view as long as it has **the same center of projection**.

Source: Alyosha Efros
…Or: Planar scene (or far away)

PP3 is a projection plane of both centers of projection, so we are OK!

This is how big aerial photographs are made

Source: Alyosha Efros
RANSAC for estimating homography

- RANSAC loop:
  1. Select four feature pairs (at random)
  2. Compute homography $H$ (exact)
  3. Compute inliers where $SSD(p_i', Hp_i) < \varepsilon$
  4. Keep largest set of inliers
  5. Re-compute least-squares $H$ estimate on all of the inliers
Robust feature-based alignment
Robust feature-based alignment

- Extract features
Robust feature-based alignment

• Extract features
• Compute *putative matches*

Source: L. Lazebnik
Robust feature-based alignment

- Extract features
- Compute *putative matches*
- Loop:
  - Hypothesize transformation $T$ (small group of putative matches that are related by $T$)

Source: L. Lazebnik
Robust feature-based alignment

• Extract features
• Compute _putative matches_
• Loop:
  – _Hypothesize_ transformation \( T \) (small group of putative matches that are related by \( T \))
  – _Verify_ transformation (search for other matches consistent with \( T \))

Source: L. Lazebnik
Robust feature-based alignment

- Extract features
- Compute *putative matches*
- Loop:
  - *Hypothesize* transformation $T$ (small group of putative matches that are related by $T$)
  - *Verify* transformation (search for other matches consistent with $T$)

Source: L. Lazebnik
Summary: alignment & warping

• Write 2d transformations as matrix-vector multiplication (including translation when we use homogeneous coordinates)

• Fitting transformations: solve for unknown parameters given corresponding points from two views (affine, projective (homography)).

• Perform image warping (forward, inverse)

• Mosaics: uses homography and image warping to merge views taken from same center of projection.
Next time: which features should we match?

Slide credit: Kristen Grauman
Questions?