CS 2770: Intro to Computer Vision

Multiple Views

Prof. Adriana Kovashka
University of Pittsburgh
March 14, 2017
Plan for today

• Affine and projective image transformations
  – Homographies and image mosaics

• Stereo vision
  – Epipolar geometry
Why multiple views?

• Structure and depth are inherently ambiguous from single views.

• Multiple views help us to perceive 3d shape and depth.
Alignment problem

• We previously discussed how to match features across images, of the same or different objects.
• Now let’s focus on the case of “two images of the same object” (e.g. $x_i$ and $x_i'$).
• What transformation relates $x_i$ and $x_i'$?
• In alignment, we will fit the parameters of some transformation according to a set of matching feature pairs (“correspondences”).

Adapted from Kristen Grauman and Derek Hoiem.
Two 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?

Kristen Grauman
Motivation: Image mosaics
Compare content in local patches, find best matches.

*Simplest approach:*

Scan $x_i'$ with template formed from a point in $x_i$, and compute Euclidean distance (a.k.a. SSD) or normalized cross-correlation between list of pixel intensities in the patch.
What are the transformations?

Examples of transformations:

- Translation
- Rotation
- Aspect
- Affine
- Perspective

Alyosha Efros
Parametric (global) warping

Transformation $T$ is a coordinate-changing machine:

$$p' = T(p)$$

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

- It is the same for any point $p$
- It can be described by just a few numbers (parameters)

Let’s represent $T$ as a matrix:

$$p' = Mp$$

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

Scaling a coordinate means multiplying each of its components by a scalar.

Uniform scaling means this scalar is the same for all components.
Scaling

*Non-uniform scaling*: different scalars per component

\[ X \times 2, \quad Y \times 0.5 \]
Scaling

Scaling operation:

\[ x' = ax \]

\[ y' = by \]

Or, in matrix form:

\[
\begin{bmatrix}
  x' \\
  y'
\end{bmatrix} =
\begin{bmatrix}
  a & 0 \\
  0 & b
\end{bmatrix}
\begin{bmatrix}
  x \\
  y
\end{bmatrix}
\]

scaling matrix \( S \)

Adapted from Alyosha Efros
2D Linear transformations

\[
\begin{bmatrix}
x'
\end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}
\]

Only linear 2D transformations can be represented with a 2x2 matrix.

Linear transformations are combinations of …

- Scale,
- Rotation,
- Shear, and
- Mirror
What transforms can we write w/ 2x2 matrix?

2D Scaling?
\[
x' = s_x \cdot x \\
y' = s_y \cdot y
\]

2D Rotate around (0,0)?
\[
x' = \cos \Theta \cdot x - \sin \Theta \cdot y \\
y' = \sin \Theta \cdot x + \cos \Theta \cdot y
\]

2D Shear?
\[
x' = x + sh_x \cdot y \\
y' = sh_y \cdot x + y
\]
What transforms can we write w/ 2x2 matrix?

2D Mirror about Y axis?

\[
x' = -x \\
y' = y
\]

\[
\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}
\]

2D Mirror over (0,0)?

\[
x' = -x \\
y' = -y
\]

\[
\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}
\]

2D Translation?

\[
x' = x + t_x \\
y' = y + t_y
\]

CAN'T DO!
Homogeneous coordinates

To convert to homogeneous coordinates:

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

homogeneous image coordinates

Converting *from* homogeneous coordinates

\[
\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow \left(\frac{x}{w}, \frac{y}{w}\right)
\]
Translation

Homogeneous Coordinates

\[
\begin{bmatrix}
  x'' \\
  y'' \\
  1
\end{bmatrix}
= \begin{bmatrix}
  1 & 0 & t_x \\
  0 & 1 & t_y \\
  0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  1
\end{bmatrix}
= \begin{bmatrix}
  x + tt_x \\
  y + tt_y \\
  1
\end{bmatrix}
\]

\[t_x = 2\]
\[t_y = 1\]
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

Maps lines to lines, parallel lines remain parallel

Adapted from Alyosha Efros
Fitting an affine transformation

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

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

\[
\begin{bmatrix}
    x_i & y_i & 0 & 0 & 1 & 0 \\
    0 & 0 & x_i & y_i & 0 & 1 \\
    \ldots & \ldots & \ldots \\
\end{bmatrix}
\begin{bmatrix}
    m_1 \\
    m_2 \\
    m_3 \\
    m_4 \\
    t_1 \\
    t_2 \\
\end{bmatrix}
= \begin{bmatrix}
    \ldots \\
    x'_i \\
    y'_i \\
    \ldots \\
\end{bmatrix}
\]

• How many matches (correspondence pairs) do we need to solve for the transformation parameters?
• Once we have solved for the parameters, how do we compute \((x'_{new}, y'_{new})\) given \((x_{new}, y_{new})\)?
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
Projective transformations

A projective transformation is a mapping between any two projective planes with *the same center of projection*

Also called **Homography**

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

Adapted from Alyosha Efros
Obtain a wider angle view by combining multiple images.
Image mosaics: Camera setup

Two images with camera rotation but no translation

Adapted from Derek Hoiem
Image mosaics: Many 2D views, one 3D object

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*
How to stitch together panorama (mosaic)?

Basic Procedure

• Take a sequence of images from the same position
  – Rotate the camera about its optical center

• Compute the homography (transformation) between first and second image

• Transform the second image to overlap with the first

• Blend the two together to create a mosaic

• (If there are more images, repeat)

Adapted from Steve Seitz
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...
Computing the homography

Assume we have four matched points:

How do we compute homography $H$?

$$
\begin{bmatrix}
wx' \\
wy' \\
w
\end{bmatrix}
= H
\begin{bmatrix}
w \\
wy' \\
w
\end{bmatrix}
= H \begin{bmatrix}
1 \\
0 \\
0
\end{bmatrix}
= \begin{bmatrix}
h_1 & h_2 & h_3 \\
h_4 & h_5 & h_6 \\
h_7 & h_8 & h_9
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
1
\end{bmatrix}
= \begin{bmatrix}
9 \\
8 \\
7 \\
6 \\
5 \\
4 \\
3 \\
2 \\
1
\end{bmatrix}
= \begin{bmatrix}
h_1 \\
h_2 \\
h_3 \\
h_4 \\
h_5 \\
h_6 \\
h_7 \\
h_8 \\
h_9
\end{bmatrix}

Using $x' = wx'/w$, $y' = wy'/w$:

Can set scale factor $h_9 = 1$.

So, there are 8 unknowns.

Need at least 8 eqs, but the more the better…

$$
\begin{bmatrix}
-x & -y & -1 & 0 & 0 & 0 & xx' & yy' & x' \\
0 & 0 & 0 & -x & -y & -1 & xy' & yy' & y'
\end{bmatrix}
\begin{bmatrix}
h_1 \\
h_2 \\
h_3 \\
h_4 \\
h_5 \\
h_6 \\
h_7 \\
h_8 \\
h_9
\end{bmatrix}
= 0

\text{Adapted from Derek Hoiem, Kristen Grauman}
How to stitch together panorama (mosaic)?

Basic Procedure

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

Adapted from Steve Seitz
Transforming the second image

\[
(x, y) \text{ \rightarrow \hspace{1cm} Image 2 \hspace{1cm} Image 1 canvas}
\]

\[
\begin{pmatrix} wx' \\
wy' \\
w \\
p' \end{pmatrix} = \begin{pmatrix} * & * & * \\
* & * & * \\
* & * & * \\
* & * & * \\
\end{pmatrix} \begin{pmatrix} x \\
y \\
1 \\
p \end{pmatrix}
\]

To apply a given homography \( H \)

- Compute \( p' = Hp \) (regular matrix multiply)
- Convert \( p' \) from homogeneous to image coordinates
Transforming the second image

Forward warping:
Send each pixel \( f(x,y) \) to its corresponding location \( (x',y') = H(x,y) \) in the right image

Modified from Alyosha Efros
Transforming the second image

Forward warping:
Send each pixel $f(x,y)$ to its corresponding location

$$(x',y') = H(x,y)$$
in the right image

Q: what if pixel lands “between” two pixels?
A: distribute color among neighboring pixels ($x',y'$)
Transforming the second image

Inverse warping:
Get each pixel $g(x',y')$ from its corresponding location $(x,y) = H^{-1}(x',y')$ in the left image
Transforming the second image

Inverse warping:
Get each pixel $g(x',y')$ from its corresponding location $(x,y) = H^{-1}(x',y')$ in the left image

Q: what if pixel comes from “between” two pixels?
A: *interpolate* color value from neighbors
Homography example: Image rectification

To unwarp (rectify) an image solve for homography $H$ given $p$ and $p'$: $p' = Hp$
Summary of affine/projective transforms

• 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 – linear, affine, projective (homography)

• **Mosaics**: uses homography and image warping to merge views taken from same center of projection
  • Perform *image warping* (forward, inverse)

Adapted from Kristen Grauman
Next topic: Stereo vision

- Homography: Same camera center, but camera rotates
- Stereo vision: Camera center is not the same (we have multiple cameras)

- Epipolar geometry
  - Relates cameras from two positions/cameras

- Stereo depth estimation
  - Recover depth from disparities between two images
Stereo photography and stereo viewers

Take two pictures of the same subject from two slightly different viewpoints and display so that each eye sees only one of the images.

Invented by Sir Charles Wheatstone, 1838

Image from fisher-price.com
Stereo photography and stereo viewers

http://www.johnsonshawmuseum.org
Depth from stereo for computers

Two cameras, simultaneous views

Single moving camera and static scene

Kristen Grauman
Depth from stereo

- Goal: recover depth by finding image coordinate $x'$ that corresponds to $x$
Depth from stereo

- Goal: recover depth by finding image coordinate $x'$ that corresponds to $x$
- Sub-Problems
  1. Calibration: How do we recover the relation of the cameras (if not already known)?
  2. Correspondence: How do we search for the matching point $x'$?
Geometry for a simple stereo system

- Assume parallel optical axes, known camera parameters (i.e., calibrated cameras). **What is expression for Z?**

![Diagram](http://www.compslab.org/~kristen/Grauman/2Dgeometry.jpg)

**Similar triangles** $(p_l, P, p_r)$ and $(O_l, P, O_r)$:

\[
\frac{T + x_l - x_r}{Z - f} = \frac{T}{Z}
\]

**Depth** is inversely proportional to **disparity**.

Depth is inversely proportional to disparity.

Adapted from Kristen Grauman
Depth from disparity

We have two images taken from cameras with different intrinsic and extrinsic parameters.

- How do we match a point in the first image to a point in the second?

If we could find the **corresponding points** in two images, we could **estimate relative depth**…
Stereo correspondence constraints

- Given $p$ in left image, where can corresponding point $p'$ be?
Epipolar constraint

Geometry of two views constrains where the corresponding pixel for some image point in the first view must occur in the second view.

- It must be on the line where (1) the plane connecting the world point and optical centers, and (2) the image plane, intersect.
- Potential matches for \( p \) have to lie on the corresponding line \( l' \).
- Potential matches for \( p' \) have to lie on the corresponding line \( l \).

Adapted from Kristen Grauman, Derek Hoiem
• **Baseline** – line connecting the two camera centers

• **Epipoles**
  = intersections of baseline with image planes
  = projections of the other camera center

• **Epipolar Plane** – plane containing baseline

• **Epipolar Lines** - intersections of epipolar plane with image planes (always come in corresponding pairs)

Adapted from Derek Hoiem
The epipolar constraint is useful because it reduces the correspondence problem to a 1D search along an epipolar line.
Stereo geometry, with calibrated cameras

If the stereo rig is calibrated, we know:
how to rotate and translate camera reference frame 1 to get to camera reference frame 2.

Rotation: 3x3 matrix $R$; translation: 3x1 vector $T$. 
If the stereo rig is calibrated, we know:
how to **rotate** and **translate** camera reference frame 1 to get to camera reference frame 2.  

\[ \mathbf{X}' = R\mathbf{X} + \mathbf{T} \]  

Adapted from Kristen Grauman
An aside: cross product

\[ \vec{a} \times \vec{b} = \vec{c} \quad \vec{a} \cdot \vec{c} = 0 \quad \vec{b} \cdot \vec{c} = 0 \]

Vector cross product takes two vectors and returns a third vector that’s perpendicular to both inputs.

So here, \( \vec{c} \) is perpendicular to both \( \vec{a} \) and \( \vec{b} \), which means the dot product equals 0.
From geometry to algebra

\[ X' = RX + T \]

\[ T \times X' = \begin{cases} \text{Normal to the plane} \\ \text{Cross-product of vector with itself is 0.} \end{cases} \]

\[ = T \times RX \]

\[ X' \cdot (T \times X') = X' \cdot (T \times RX) = 0 \]
Aside: Matrix form of cross product

\[ \vec{a} \times \vec{b} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \vec{c} \]

Can be expressed as a matrix multiplication.

\[ \vec{a} \cdot \vec{c} = 0 \]
\[ \vec{b} \cdot \vec{c} = 0 \]
Essential matrix

\[ X' \cdot (T \times RX) = 0 \]

\[ X' \cdot ([T_x]RX) = 0 \]

Let \( E = [T_x]R \)

\[ X' \cdot EX = X'^TEX = 0 \]

\( E \) is called the essential matrix, and it relates corresponding image points between both cameras, given the rotation and translation.

Before we said: If we observe a point in one image, its position in other image is constrained to lie on line defined by above. It turns out that:

- \( E^T x \) is the epipolar line \( l' \) through \( x' \) in the second image, corresponding to \( x \).
- \( Ex' \) is the epipolar line \( l \) through \( x \) in the first image, corresponding to \( x' \).
Essential matrix example: parallel cameras

For the parallel cameras, image of any point must lie on same horizontal line in each image plane.

Kristen Grauman
image $I(x,y)$

Disparity map $D(x,y)$

image $I'(x',y')$

$(x',y') = (x + D(x,y), y)$

Adapted from Kristen Grauman
Basic stereo matching algorithm

- For each pixel in the first image
  - Find corresponding epipolar scanline in the right image
  - Search along epipolar line and pick the best match $x'$
    - Slide a window along the right scanline and compute Euclidean distance between contents of that window with the reference window in the left image; take the window corresponding to the minimum as the match
  - Compute disparity $x - x'$ and set $\text{depth}(x) = f*T/(x-x')$

Adapted from Derek Hoiem
Results with window search

Data

Window-based matching

Ground truth
Summary of stereo vision

• **Epipolar geometry**
  – Epipoles are intersection of baseline with image planes
  – Matching point in second image is on a line passing through its epipole
  – Epipolar constraint limits where points from one view will be imaged in the other, which makes search for correspondences quicker
  – Essential matrix $E$ maps from a point in one image to a line (its epipolar line) in the other

• **Stereo depth estimation**
  – Find corresponding points along epipolar scanline
  – Estimate disparity (depth is inverse to disparity)

Adapted from Kristen Grauman and Derek Hoiem
How can we improve?

• Uniqueness
  – For any point in one image, there should be at most one matching point in the other image

• Ordering
  – Corresponding points should be in the same order in both views

• Smoothness
  – We expect disparity values to change slowly (for the most part)

Many of these constraints can be encoded in an energy function and solved using graph cuts

For the latest and greatest: [http://vision.middlebury.edu/stereo/](http://vision.middlebury.edu/stereo/)
Projective structure from motion

- Given: $m$ images of $n$ fixed 3D points
  \[ x_{ij} = P_i X_j, \quad i = 1, \ldots, m, \quad j = 1, \ldots, n \]
- Problem: estimate $m$ projection matrices $P_i$ and $n$ 3D points $X_j$ from the $mn$ corresponding 2D points $x_{ij}$
Photo tourism


http://phototour.cs.washington.edu/
3D from multiple images