CS5670: Computer Vision

Image transformations and image warping
Reading

• Szeliski (1st edition): Chapter 3.6
Announcements

• Project 2 out, due Friday, March 12 by 7pm
  – Do be done in groups of 2 – if you need help finding a partner, try Ed Discussions or let us know
Image alignment

Why don’t these image line up exactly?
What is the geometric relationship between these two images?

Answer: Similarity transformation (translation, rotation, uniform scale)
What is the geometric relationship between these two images?
What is the geometric relationship between these two images?

Very important for creating mosaics!
First, we need to know what this transformation is. Second, we need to figure out how to compute it using feature matches.
Image Warping

• image filtering: change range of image
  
  \[ g(x) = h(f(x)) \]

• image warping: change domain of image

\[ g(x) = f(h(x)) \]
Image Warping

• image filtering: change *range* of image
  - $g(x) = h(f(x))$

• image warping: change *domain* of image
  - $g(x) = f(h(x))$
Parametric (global) warping

• Examples of parametric warps:

  - translation
  - rotation
  - aspect
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 consider linear xforms (can be represented by a 2x2 matrix):
  \[ p' = Tp \]
  \[ \begin{bmatrix} x' \\ y' \end{bmatrix} = T \begin{bmatrix} x \\ y \end{bmatrix} \]
Common linear transformations

- Uniform scaling by $s$:

\[
S = \begin{bmatrix}
s & 0 \\
0 & s \\
\end{bmatrix}
\]

What is the inverse?
Common linear transformations

- Rotation by angle $\theta$ (about the origin)

$$\mathbf{R} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$$

What is the inverse?
For rotations:
$$\mathbf{R}^{-1} = \mathbf{R}^T$$
2x2 Matrices

• What types of transformations can be represented with a 2x2 matrix?

2D mirror across Y axis?

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

2D mirror across line \( y = x \)?

\[ x' = y \]
\[ y' = x \]
2x2 Matrices

- What types of transformations can be represented with a 2x2 matrix?

2D mirror across Y axis?

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

\[
T = \begin{bmatrix}
-1 & 0 \\
0 & 1 \\
\end{bmatrix}
\]

2D mirror across line y = x?

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

\[
T = \begin{bmatrix}
0 & 1 \\
1 & 0 \\
\end{bmatrix}
\]
2x2 Matrices

• What types of transformations can be represented with a 2x2 matrix?

2D Translation?

\[ x' = x + t_x \]
\[ y' = y + t_y \]
2x2 Matrices

- What types of transformations can be represented with a 2x2 matrix?

**2D Translation?**

\[
\begin{align*}
    x' &= x + t_x \\
    y' &= y + t_y
\end{align*}
\]

NO!

Translation is not a linear operation on 2D coordinates
All 2D Linear Transformations

• Linear transformations are combinations of ...
  – Scale, \( \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \)
  – Rotation,
  – Shear, and
  – Mirror

• Properties of linear transformations:
  – Origin maps to origin
  – Lines map to lines
  – Parallel lines remain parallel
  – Ratios are preserved
  – Closed under composition

\[
\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b & e & f \\ c & d & g & h \\ i & j & k & l \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}
\]
Homogeneous coordinates

Trick: add one more coordinate:

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

homogeneous image coordinates

Converting from homogeneous coordinates

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow \left(\frac{x}{w}, \frac{y}{w}, 1\right)$$
Translation

- Solution: homogeneous coordinates to the rescue

\[
T = \begin{bmatrix}
1 & 0 & t_x \\
0 & 1 & t_y \\
0 & 0 & 1
\end{bmatrix}
\]

\[
\begin{bmatrix}
1 & 0 & t_x \\
0 & 1 & t_y \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
1
\end{bmatrix} =
\begin{bmatrix}
x + t_x \\
y + t_y \\
1
\end{bmatrix}
\]
Affine transformations

\[
T = \begin{bmatrix}
1 & 0 & t_x \\
0 & 1 & t_y \\
0 & 0 & 1
\end{bmatrix}
\]

any transformation represented by a 3x3 matrix with last row \([0 \ 0 \ 1]\) we call an \textit{affine} transformation

\[
\begin{bmatrix}
a & b & c \\
d & e & f \\
0 & 0 & 1
\end{bmatrix}
\]
Basic affine transformations

Translate

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

Scale

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

2D in-plane rotation

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

Shear

\[
\begin{bmatrix}
x' \\
y' \\
1
\end{bmatrix} =
\begin{bmatrix}
1 & sh_x & 0 \\
sh_y & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
1
\end{bmatrix}
\]
Affine transformations

• Affine transformations are combinations of ...
  – Linear transformations, and
  – Translations

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

• Properties of affine transformations:
  – Origin does not necessarily map to origin
  – Lines map to lines
  – Parallel lines remain parallel
  – Ratios are preserved
  – Closed under composition
Is this an affine transformation?
Where do we go from here?

affine transformation

\[
\begin{bmatrix}
a & b & c \\
d & e & f \\
0 & 0 & 1 \\
\end{bmatrix}
\]

what happens when we mess with this row?
Projective Transformations *aka* Homographies *aka* Planar Perspective Maps

\[
H = \begin{bmatrix}
  a & b & c \\
  d & e & f \\
  g & h & 1 \\
\end{bmatrix}
\]

Called a *homography* (or *planar perspective map*)
Homographies

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

What happens when the denominator is 0?

\[
\begin{bmatrix}
\frac{ax+by+c}{gx+hy+1} \\
\frac{dx+ey+f}{gx+hy+1} \\
1
\end{bmatrix}
\]
Points at infinity
Image warping with homographies

image plane in front

black area where no pixel maps to
Homographies
Homographies

• Homographies ...
  – Affine transformations, and
  – Projective warps

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

• Properties of projective transformations:
  – Origin does not necessarily map to origin
  – Lines map to lines
  – Parallel lines do not necessarily remain parallel
  – Ratios are not preserved
  – Closed under composition
Alternate formulation for homographies

\[
\begin{bmatrix}
  x'_i \\
  y'_i \\
  1
\end{bmatrix} \mapsto \begin{bmatrix}
  h_{00} & h_{01} & h_{02} \\
  h_{10} & h_{11} & h_{12} \\
  h_{20} & h_{21} & h_{22}
\end{bmatrix} \begin{bmatrix}
  x_i \\
  y_i \\
  1
\end{bmatrix}
\]

where the length of the vector \([h_{00} \ h_{01} \ ... \ h_{22}]\) is 1.
2D image transformations

These transformations are a nested set of groups
• Closed under composition and inverse is a member
Implementing image warping

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

• Send each pixel \((x,y)\) to its corresponding location \((x',y')\) = \(T(x,y)\) in \(g(x',y')\)

  • What if pixel lands “between” two pixels?
Forward Warping

• Send each pixel \((x, y)\) to its corresponding location \((x', y')\)
  
  \[ T(x, y) \text{ in } g(x', y') \]

  • What if pixel lands “between” two pixels?
  • Answer: add “contribution” to several pixels, normalize later (splatting)

  • Can still result in holes

\[ f(x, y) \text{ and } g(x', y') \]
Inverse Warping

• Get each pixel $g(x',y')$ from its corresponding location $(x,y) = T^{-1}(x,y)$ in $f(x,y)$
  • Requires taking the inverse of the transform
  • 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 $f(x,y)$
  • What if pixel comes from “between” two pixels?
  • Answer: resample color value from interpolated (prefiltered) source image
Interpolation

• Possible interpolation filters:
  – nearest neighbor
  – bilinear
  – bicubic
  – sinc

• Needed to prevent “jaggies” and “texture crawl”
  (with prefiltering)
Questions?