E1 216 COMPUTER VISION

LECTURE 07: GEOMETRIC TRANSFORMATIONS

Venu Madhav Govindu
Department of Electrical Engineering
Indian Institute of Science, Bengaluru

2023

Geometric Transformations

- Use multiple or single image(s)
- Geometric - pure 3D rotations - mosaics
- Radiometric - high dynamic range imaging
- Focus on **geometric** transformations

Geometric Transformations

coolopticalillusions.com

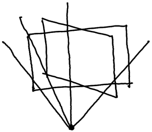
Geometric Transformations

coolopticalillusions.com

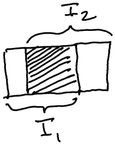
$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \boldsymbol{K} \left[\begin{array}{c|c} \boldsymbol{R} & \boldsymbol{T} \end{array} \right] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Pinhole Camera

- Effects of rotations and translations are mixed
- **Only rotations ? (Mosaics)**
- Only translations ? (Stereo; considered later)
- Both ? (Multiview Geometry; considered later)



$$p_1 = \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = \boldsymbol{K} \left[\; \boldsymbol{I} \mid \boldsymbol{0} \; \right] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \boldsymbol{KP}$$

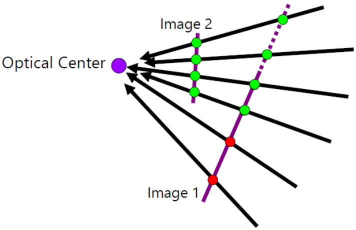


$$p_2 = \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = \boldsymbol{K} \left[\; \boldsymbol{R} \mid \boldsymbol{0} \; \right] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \boldsymbol{KRP}$$

$$\boxed{p_2 = \boldsymbol{KRK}^{-1} p_1}$$

Pure 3D Camera Rotation

- $\boldsymbol{P} = [X, Y, Z]^T$
- Pure 3D Rotations is a special case
- p_1 and p_2
  - related via camera parameters
  - does not depend on 3D geometry



Rotating Camera

- Centre of projection same for all cameras
- Each image samples from same parametric ray set
- No "parallax" problem
- Depth plays no role
- Excellent for mosaics
- Equivalent to wider FOV camera

From Noah Snavely's slides

Why Mosaic?

Are you getting the whole picture?

- Compact Camera FOV = 50 x 35°

Why Mosaic?

Are you getting the whole picture?

- Compact Camera FOV = 50 x 35°
- Human FOV = 200 x 135°

Slide from Brown & Lowe

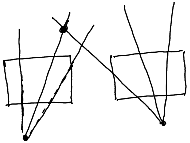
Why Mosaic?

Are you getting the whole picture?

- Compact Camera FOV = 50 x 35°
- Human FOV = 200 x 135°
- Panoramic Mosaic = 360 x 180°

Slide from Brown & Lowe

Geometric Transformations



$$p_1 = \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = \boldsymbol{K} \left[\ \boldsymbol{I} \mid \mathbf{0} \ \right] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$p_2 = \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = \boldsymbol{K} \left[\ \boldsymbol{I} \mid \mathbf{T} \ \right] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$\boldsymbol{p}_1 = \boldsymbol{K}\boldsymbol{P}$ and $\boldsymbol{p}_2 = \boldsymbol{K}(\boldsymbol{P} + \boldsymbol{T})$

$$x_2 - x_1 = \frac{fB}{Z}$$

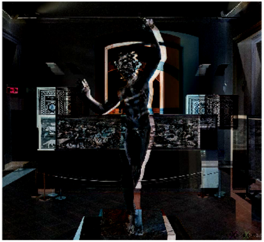
Pure 3D Camera Translation

- $\boldsymbol{P} = [X, Y, Z]^T$
- $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ related via translation and depth
- No simple relationship like pure rotations
- Used to recover 3D depth (stereo)

Geometric Transformations

urixblog.com

Geometric Transformations



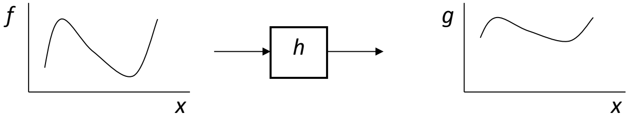
Pure 3D Translations

- No single geometric (parametric) transformation
- Non-linear dependence on depth
- Use to estimate depth (stereo)
- Effects of 3D rotation and translation are complementary

We can also take a purely 2D geometric transformation view
Following slides borrowed from Noah Snavely

Image Warping

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



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

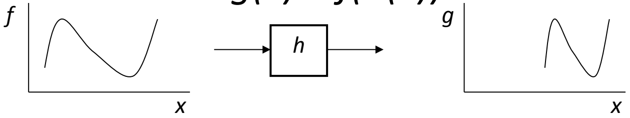
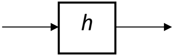


Image Warping

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

f



g

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

f

g

Parametric (global) warping

- Examples of parametric warps:

translation

rotation

aspect

affine

perspective

cylindrical

Parametric (global) warping

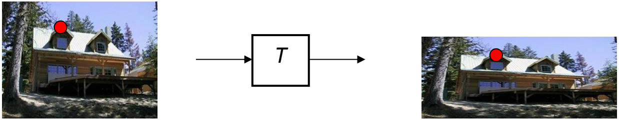
- Examples of parametric warps:

translation

rotation

aspect

Parametric (global) warping



p = (x,y) **p’** = (x’,y’)

- 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 2D matrix):

$$\mathbf{p}' = \mathbf{T}\mathbf{p} \qquad \begin{bmatrix} x' \\ y' \end{bmatrix} = \mathbf{T} \begin{bmatrix} x \\ y \end{bmatrix}$$

Common linear transformations

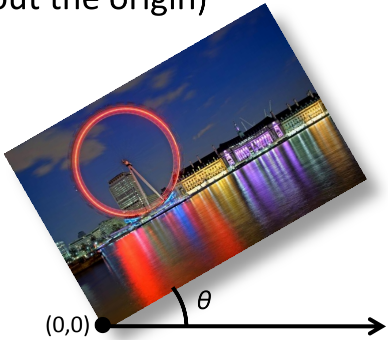
- Uniform scaling by *s*:

$$\mathbf{S} = \begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix}$$

What is the inverse?

Common linear transformations

- Rotation by angle θ (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 about Y axis?

$$\begin{aligned} x' &= -x \\ y' &= y \end{aligned} \qquad \mathbf{T} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

2D mirror across line y = x?

$$\begin{aligned} x' &= y \\ y' &= x \end{aligned} \qquad \mathbf{T} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

2x2 Matrices

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

2D Translation?

$$\begin{aligned} x' &= x + t_x \\ y' &= y + t_y \end{aligned}$$

NO!

$\boxed{\text{Translation is not a linear operation on 2D coordinates}}$

All 2D Linear Transformations

- Linear transformations are combinations of …
  - Scale,
  - Rotation,
  - Shear, and
  - Mirror

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

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

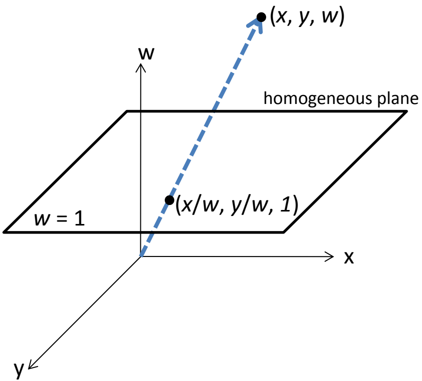
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} e & f \\ g & h \end{bmatrix} \begin{bmatrix} i & j \\ k & l \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Homogeneous coordinates

Trick: add one more coordinate:

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

homogeneous image coordinates



Converting *from* homogeneous coordinates

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

Translation

- Solution: homogeneous coordinates to the rescue

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x + t_x \\ y + t_y \\ 1 \end{bmatrix}$$

Affine transformations

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

any transformation with last row [0 0 1] we call an *affine* transformation

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

Basic affine transformations

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

Translate

$$\begin{bmatrix} \boldsymbol{x}' \\ \boldsymbol{y}' \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{s}_x & 0 & 0 \\ 0 & \boldsymbol{s}_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \\ 1 \end{bmatrix}$$

Scale

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

2D *in-plane* rotation

$$\begin{bmatrix} \boldsymbol{x}' \\ \boldsymbol{y}' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & \boldsymbol{sh}_x & 0 \\ \boldsymbol{sh}_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \\ 1 \end{bmatrix}$$

Shear

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

Where do we go from here?

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

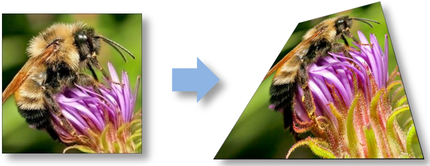
what happens when we mess with this row?

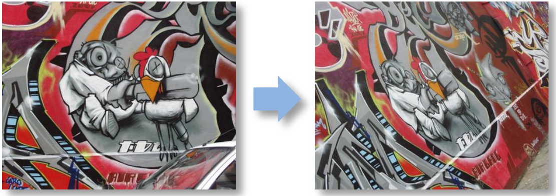
affine transformation

Projective Transformations aka Homographies aka Planar Perspective Maps

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

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





Homographies

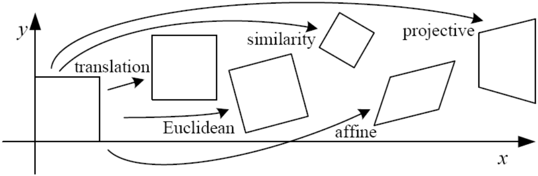
Homographies

- 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

2D image transformations



Name	Matrix	# D.O.F.	Preserves:	Icon
translation	$\begin{bmatrix} \boldsymbol{I} \mid \boldsymbol{t} \end{bmatrix}_{2\times3}$	2	orientation $+\cdots$	
rigid (Euclidean)	$\begin{bmatrix} \boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times3}$	3	lengths $+\cdots$	
similarity	$\begin{bmatrix} s\boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times3}$	4	angles $+\cdots$	
affine	$\begin{bmatrix} \boldsymbol{A} \end{bmatrix}_{2\times3}$	6	parallelism $+\cdots$	
projective	$\begin{bmatrix} \tilde{\boldsymbol{H}} \end{bmatrix}_{3\times3}$	8	straight lines	

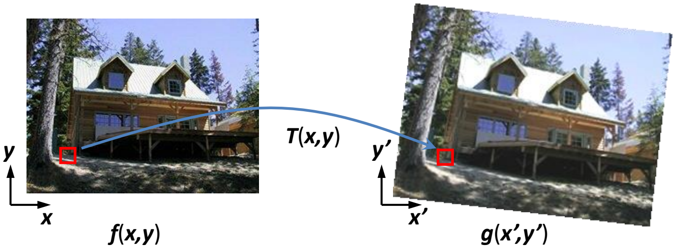
These transformations are a nested set of groups

- Closed under composition and inverse is a member

Homographies

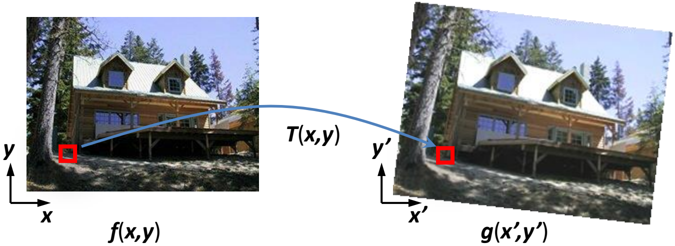
Image Warping

- Given a coordinate xform $(\boldsymbol{x'},\boldsymbol{y'}) = \boldsymbol{T}(\boldsymbol{x},\boldsymbol{y})$ and a source image $\boldsymbol{f}(\boldsymbol{x},\boldsymbol{y})$, how do we compute an xformed image $\boldsymbol{g}(\boldsymbol{x'},\boldsymbol{y'}) = \boldsymbol{f}(\boldsymbol{T}(\boldsymbol{x},\boldsymbol{y}))$?



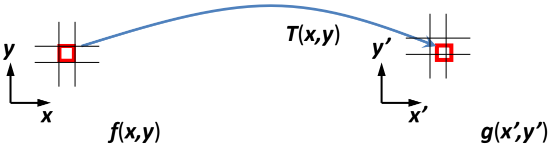
Forward Warping

- Send each pixel $f(x)$ to its corresponding location $(x',y') = T(x,y)$ in $g(x',y')$
  - What if pixel lands "between" two pixels?



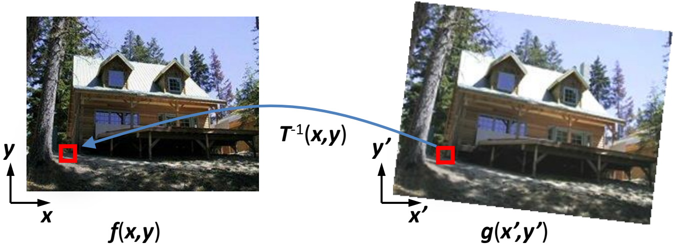
Forward Warping

- Send each pixel $\boldsymbol{f}(\boldsymbol{x},\boldsymbol{y})$ to its corresponding location $\boldsymbol{x'} = \boldsymbol{h}(\boldsymbol{x},\boldsymbol{y})$ in $\boldsymbol{g}(\boldsymbol{x'},\boldsymbol{y'})$
  - What if pixel lands "between" two pixels?
  - Answer: add "contribution" to several pixels, normalize later (*splatting*)
  - Can still result in holes



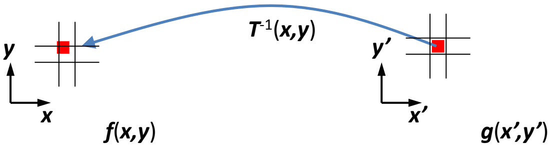
Inverse Warping

- Get each pixel $\boldsymbol{g}(\boldsymbol{x}',\boldsymbol{y}')$ from its corresponding location $(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{T}^{-1}(\boldsymbol{x},\boldsymbol{y})$ in $\boldsymbol{f}(\boldsymbol{x},\boldsymbol{y})$
  - Requires taking the inverse of the transform
  - What if pixel comes from "between" two pixels?



Inverse Warping

- Get each pixel ***g***(***x'***) from its corresponding location ***x'*** = ***h***(***x***) in ***f***(***x***)
  - What if pixel comes from “between” two pixels?
  - Answer: *resample* color value from *interpolated* (*prefiltered*) source image

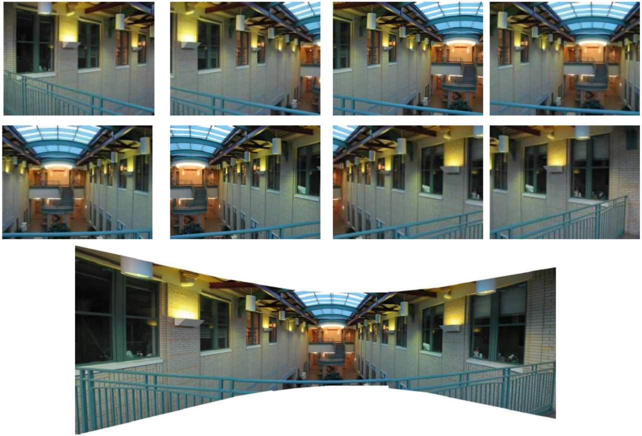


Interpolation

- Possible interpolation filters:
  - nearest neighbor
  - bilinear
  - bicubic (interpolating)
  - sinc
- Needed to prevent "jaggies" and "texture crawl"

  (with prefiltering)

Geometric Transformations

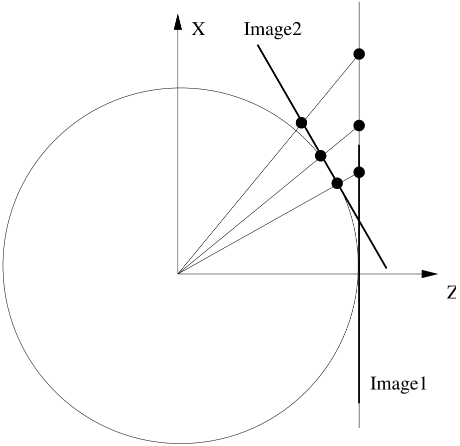


Enlarged FOV; Why do we have a radial shape ?

Following slides on impact of geometry of virtual camera plane
Taken from Magnus Oskarsson's slides

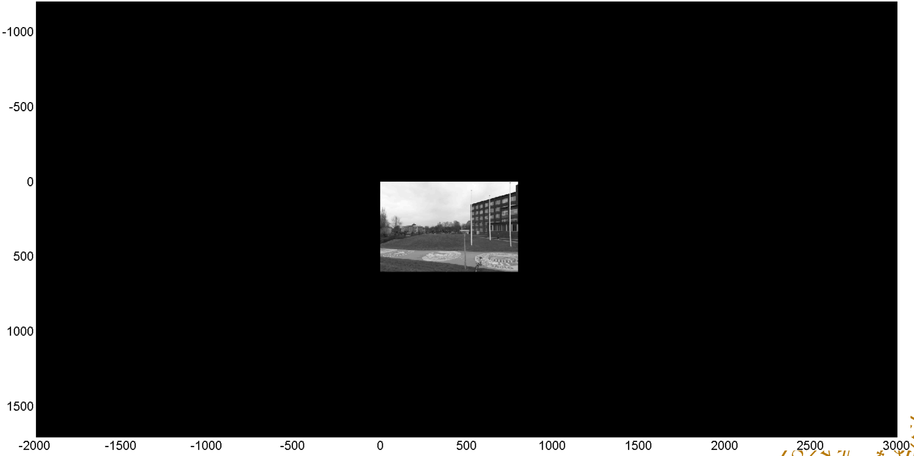
Panoramas

For calibrated cameras:

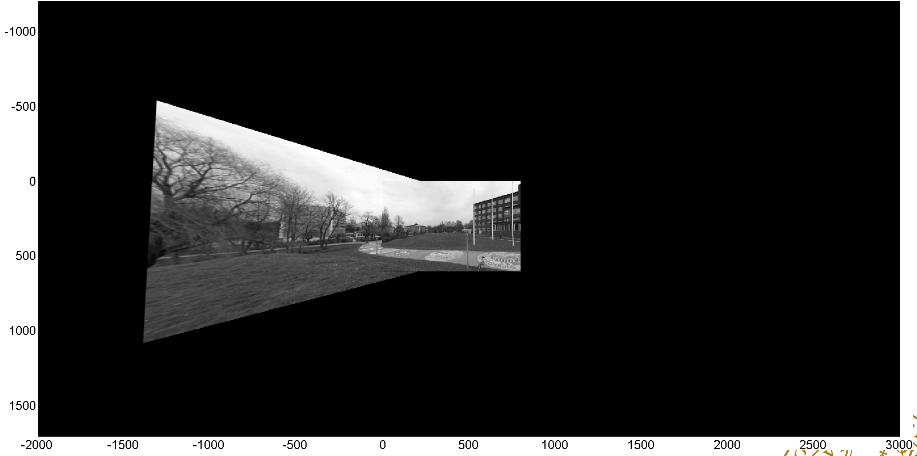


Panoramas

Panoramas

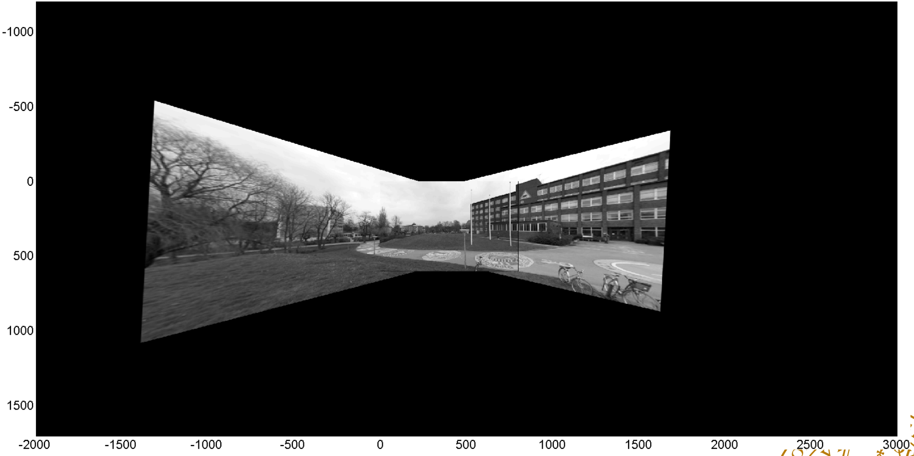

-1000
-500
0
500
1000
1500
-2000
-1500
-1000
-500
0
500
1000
1500
2000
2500
3000

Panoramas



Points are transformed to the first image.

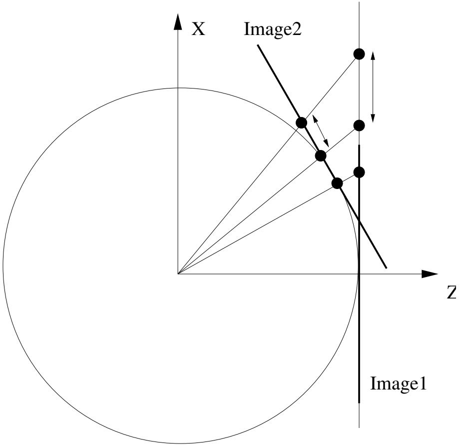
Panoramas



-1000
-500
0
500
1000
1500
-2000
-1500
-1000
-500
0
500
1000
1500
2000
2500
3000

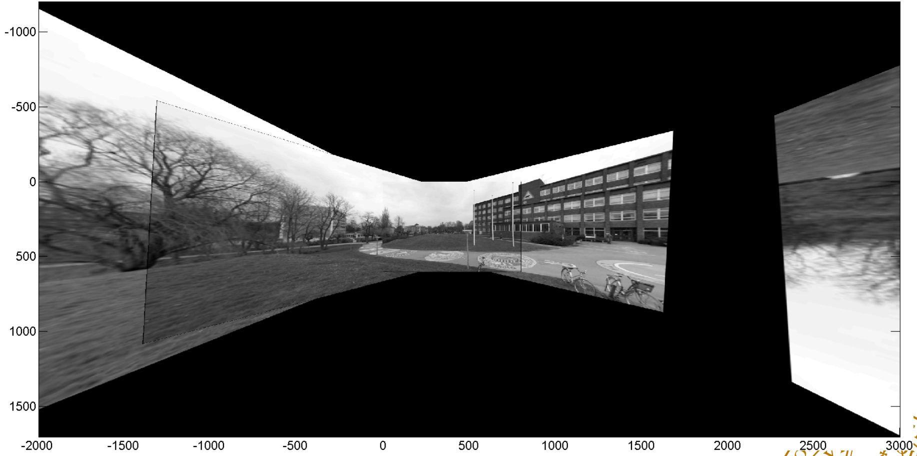
Panoramas

For calibrated cameras:



Distances are not preserved. Points close to the x-axis tend to infinity.

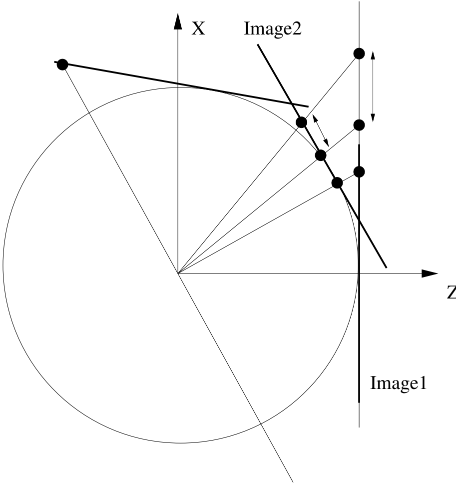
Panoramas



-1000
-500
0
500
1000
1500
-2000
-1500
-1000
-500
0
500
1000
1500
2000
2500
3000

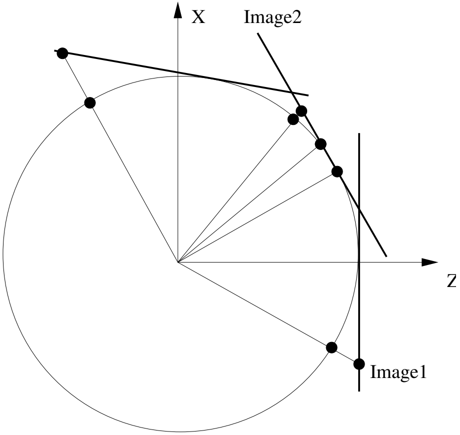
Panoramas

For calibrated cameras:



Cannot transfer all points into the first image.

For calibrated cameras:



Project onto a cylinder instead.

Panoramas

For calibrated cameras:

Distances are roughly preserved. Lines may not appear straight.

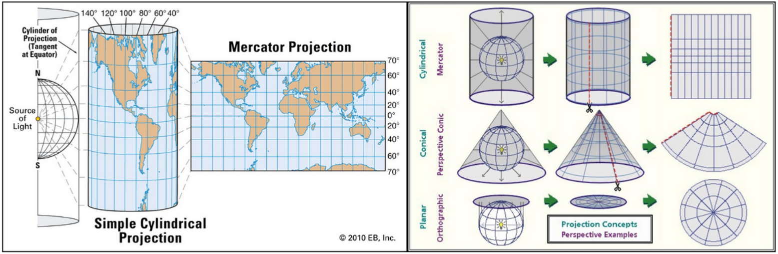


Figure 3: A simplistic model showing how Projected Coordinate Systems are created using a sphere. Source: Britannica.

- Topology of sphere $\neq$ that of 2D plane
- Issue has plagued map making!

https://medium.com/nightingale/understanding-map-projections-8b23ecbd2a2f

Figure 4: The Mercator projection exaggerates the size of the countries as you move away from the Equator. Source: snippet from The True Size Of.

Geometric Transformations

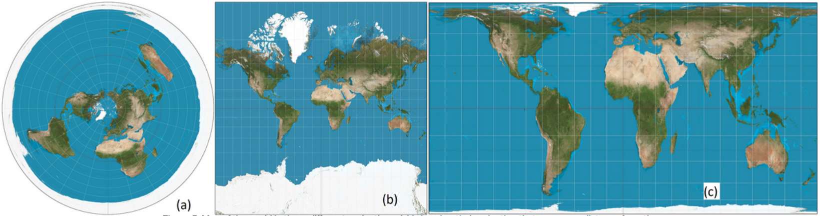


Figure 5: Map of the world in three different projections: (a) is in azimuthal projection that preserves distance from the center point, (b) is in aMercator projection that preserves shape, and (c) is in cylindrical equal-area projection that preserves area. Source: Wikipedia.

https://medium.com/nightingale/understanding-map-projections-8b23ecbd2a2f

Choosing the right projection system

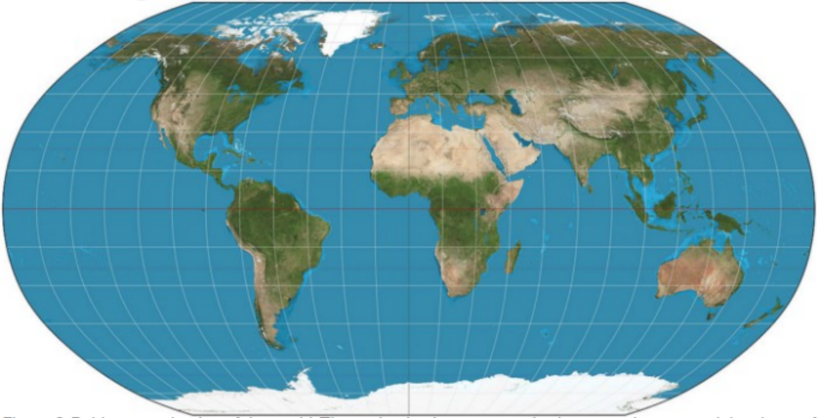


Figure 6: Robinson projection of the world. The projection is a compromise between the area and the shape of the world. Source: Wikipedia.

https://medium.com/nightingale/understanding-map-projections-8b23ecbd2a2f

AFRICA IS BIG!

Why is north ‘up’ ?

Recovering Geometry

- Recall pure 3D rotations
- $\boldsymbol{p}_2 = \boldsymbol{K}\boldsymbol{R}\boldsymbol{K}^{-1}\boldsymbol{p}_1$
- Do we need to know $\boldsymbol{K}$ and $\boldsymbol{R}$?
- $\boldsymbol{H} = \boldsymbol{K}\boldsymbol{R}\boldsymbol{K}^{-1}$
- $\boldsymbol{H}$ is 3×3 projective matrix
- $\boldsymbol{H}$ is a homography/collineation/projective transformation
- $\boldsymbol{p}_2 = \boldsymbol{H}\boldsymbol{p}_1$

Recovering Geometry

- Recall pure 3D rotations
- $\boldsymbol{p}_2 = \boldsymbol{K}\boldsymbol{R}\boldsymbol{K}^{-1}\boldsymbol{p}_1$
- Do we need to know $\boldsymbol{K}$ and $\boldsymbol{R}$?
- $\boldsymbol{H} = \boldsymbol{K}\boldsymbol{R}\boldsymbol{K}^{-1}$
- $\boldsymbol{H}$ is 3×3 projective matrix
- $\boldsymbol{H}$ is a homography/collineation/projective transformation
- $\boldsymbol{p}_2 = \boldsymbol{H}\boldsymbol{p}_1$

Geometric Transformations

Recovering Geometry

- Recall pure 3D rotations
- $\boldsymbol{p}_2 = \boldsymbol{K}\boldsymbol{R}\boldsymbol{K}^{-1}\boldsymbol{p}_1$
- Do we need to know $\boldsymbol{K}$ and $\boldsymbol{R}$?
- $\boldsymbol{H} = \boldsymbol{K}\boldsymbol{R}\boldsymbol{K}^{-1}$
- $\boldsymbol{H}$ is 3×3 projective matrix
- $\boldsymbol{H}$ is a homography/collineation/projective transformation
- $\boldsymbol{p}_2 = \boldsymbol{H}\boldsymbol{p}_1$

Homography relationship

How can we use this relationship $\boldsymbol{p}_2 = \boldsymbol{H}\boldsymbol{p}_1$

- Radiometric: $I_1(\boldsymbol{p}) = I_2(\boldsymbol{H}\boldsymbol{p})$
- Is this always true?
- Geometric: $\boldsymbol{p}_2 = \boldsymbol{H}\boldsymbol{p}_1$
- Need correspondences $\boldsymbol{p}_1 \leftrightarrow \boldsymbol{p}_2$

Geometric Transformations

$$
\begin{aligned}
\boldsymbol{H} &= \arg\min_{\boldsymbol{H}} ||I_1(\boldsymbol{p}) - I_2(\boldsymbol{H}\boldsymbol{p})||^2 \\
\text{Update step} & \quad \boldsymbol{H} \leftarrow \boldsymbol{H} + \delta\boldsymbol{H} \\
\text{Use} & \quad I_2((\boldsymbol{H} + \delta\boldsymbol{H})\boldsymbol{p}) \approx I_2(\boldsymbol{H}\boldsymbol{p}) + \boldsymbol{J}^T\delta\boldsymbol{H} \\
\text{Minimise} & \quad ||\boldsymbol{J}^T\delta\boldsymbol{H} - (I_1(\boldsymbol{p}) - I_2(\boldsymbol{H}\boldsymbol{p}))||^2
\end{aligned}
$$

Estimating Homographies

- Solution: Least square fit of intensities
- Is it a linear problem?
- Warp, Update, Warp, till convergence
- Use all pixels in overlapping area
- Robust loss $\rho(.)$ for each pixel
- Multiscale approaches used. Why?
- Many issues in estimation

Geometric Transformations

$$\begin{aligned} \boldsymbol{p}_2 &= \boldsymbol{H}\boldsymbol{p}_1 \\ \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} &= \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \end{aligned}$$

Geometric Estimation

- Correspondences $\boldsymbol{p}_1 \leftrightarrow \boldsymbol{p}_2$ (SIFT etc.)
- $\boldsymbol{p}_2 = \boldsymbol{H}\boldsymbol{p}_1$ is a projective relationship
- Non-linear relationship?

Geometric Transformations

$$\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

Implies

$$x_2 = \frac{h_{11}x_1 + h_{12}y_1 + h_{13}}{h_{31}x_1 + h_{32}y_1 + h_{33}} \quad (1)$$

$$y_2 = \frac{h_{21}x_1 + h_{22}y_1 + h_{13}}{h_{31}x_1 + h_{32}y_1 + h_{33}} \quad (2)$$

Can solve using non-linear least squares on equations

$$x_2 = \frac{h_{11}x_1 + h_{12}y_1 + h_{13}}{h_{31}x_1 + h_{32}y_1 + h_{33}}$$
$$y_2 = \frac{h_{21}x_1 + h_{22}y_1 + h_{13}}{h_{31}x_1 + h_{32}y_1 + h_{33}}$$

Linear in entries of $\boldsymbol{H}$, carry-over will result in

$$x_2(h_{31}x_1 + h_{32}y_1 + h_{33}) - (h_{11}x_1 + h_{12}y_1 + h_{13}) = 0$$
$$y_2(h_{31}x_1 + h_{32}y_1 + h_{33}) - (h_{21}x_1 + h_{22}y_1 + h_{23}) = 0$$

Leads to

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x_2 & -y_1x_2 & -x_2 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y_2 & -y_1y_2 & -y_2 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ \vdots \\ h_{33} \end{bmatrix} = \mathbf{0}$$

Geometric Transformations

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x_2 & -y_1x_2 & -x_2 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y_2 & -y_1y_2 & -y_2 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ \vdots \\ h_{33} \end{bmatrix} = \mathbf{0}$$

Linear Method

- 2 eqns per correspondence
- Unknowns in $\boldsymbol{H}$?
- Collect all equations into $\boldsymbol{Ah} = \mathbf{0}$ problem
- Solution ?
- Two important considerations
  - Robustness (RANSAC or IRLS?)
  - Conditioning (Scale of data)

Geometric Transformations

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x_2 & -y_1x_2 & -x_2 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y_2 & -y_1y_2 & -y_2 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ \vdots \\ h_{33} \end{bmatrix} = \mathbf{0}$$

Normalisation

- Recall that all correspondences are noisy
- (x, y) co-ordinates of order of 1000
- Quadratic terms in $\boldsymbol{A}$
- Errors in observed $\boldsymbol{A}$ are not uniform in dimensions
- Leads to very poor conditioning of $\boldsymbol{Ah} = \mathbf{0}$
- Remedy
  - Scale co-ordinates (x, y) to have magnitude around 1
  - Solve
  - Put back original scale

* Find features using SIFT (vl_feat package) or SURF (inbuilt in MATLAB)

https://www.vlfeat.org/overview/sift.html

* Feature matching using vl_feat or MATLAB inbuilt functions.

NOTE: PLEASE BE MINDFUL OF THE CONVENTIONS USED FOR IMAGE COORDINATE SYSTEM, CAMERA " "

* Raw (putative) matches: outliers included. Need robustness while estimating homography.

$$P' = \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} \longleftrightarrow \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = P$$ are matched features in the 2 images.

$$P' = HP.$$

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

$$x' = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}} \longrightarrow (a)$$

$$y' = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}} \longrightarrow (b)$$

Method-1:

(a): $(h_{31}x + h_{32}y + h_{33})x' = h_{11}x + h_{12}y + h_{13}$

$$\Rightarrow h_{31}xx' + h_{32}yx' + h_{33}x' - h_{11}x - h_{12}y - h_{13} = 0$$

$$(-x)h_{11} + (-y)h_{12} + (-1)h_{13} + (0)h_{21} + (0)h_{22} + (0)h_{23} + (xx')h_{31} + (yx')h_{32} + (x')h_{33} = 0 \quad (1)$$

(b):

$$(0)h_{11} + (0)h_{12} + (0)h_{13} + (-x)h_{21} + (-y)h_{22} + (-1)h_{23} + (xy')h_{31} + (yy')h_{32} + (y')h_{33} = 0 \quad (2)$$

$h_{11}, \dots\, h_{33}$ are the unknowns.

$$\underbrace{\begin{bmatrix} -x & -y & -1 & 0 & 0 & 0 & xx' & yx' & x' \\ 0 & 0 & 0 & -x & -y & -1 & xy' & yy' & y' \\ & & & \vdots & & & & & \\ & & & \vdots & & & & & \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ \vdots \\ h_{33} \end{bmatrix}}_{h} = 0$$

$$\boxed{Ah = 0} \rightarrow (1)$$

Soln: h: least eigen vector of (A^TA)

(or)

h: the right singular vector of 'A' corresponding to the least singular value.

Normalisation:

Consider $\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 400 \\ 200 \end{pmatrix}$ $\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} 800 \\ 400 \end{pmatrix}$

$$\begin{bmatrix} -4\times10^2 & -2\times10^2 & -1 & 0 & 0 & 0 & 3.2\times10^5 & 1.6\times10^5 & 8\times10^2 \\ 0 & 0 & 0 & -4\times10^2 & -2\times10^2 & -1 & 1.6\times10^5 & 0.8\times10^5 & 4\times10^2 \end{bmatrix}$$

Note that the entries are ranging from $0 - 10^5$ in A, then it will range from $0 - 10^{10}$ in A^TA.

Lack of homogenity in the coordinates $\Rightarrow$ poor conditioning of A (or) A^TA.

$\rightarrow$ We want to reduce the range of the image coordinates entries in A ($\Leftrightarrow A^TA$) to improve the conditioning (i.e, to reduce the $K(A^TA)$).

→ Apply some transformations to the image coordinates in each image.

(i) translation

(ii) scaling.

(i) Translation: Origin of the new coordinate system should be at the centroid of the image points.

(ii) After translation, the coordinates are scaled s.t mean distance from the origin to a point equals $\sqrt{2}$.

(i) $\{x_i\}, \{y_i\}, \; i = 1 \dots N.$

$$\bar{x} = \frac{1}{N}\sum_i x_i \;, \quad \bar{y} = \frac{1}{N}\sum_i \cancel{x}y_i$$

we want, $x_i' = x_i - \bar{x}$

$$\frac{1}{N}\sum_i x_i' = \frac{1}{N}\sum_i x_i - \bar{x} = 0.$$

$$\begin{bmatrix} x_i' \\ y_i' \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & -\bar{x} \\ 0 & 1 & -\bar{y} \\ 0 & 0 & 1 \end{bmatrix}}_{T_{11}} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

(ii) Mean distance from the origin to a point equals $\sqrt{2}$.

$$C' = \frac{1}{N} \sum_i d(x_i', y_i')$$

$$= \frac{1}{N} \sum_i \left(\sqrt{x_i'^2 + y_i'^2} \right)$$

Now if, $\quad x_i'' = \sqrt{2}\, x_i'/C' \; ; \quad y_i'' = \sqrt{2}\, y_i'/C'$

Then,

$$C'' = \frac{1}{N} \sum_i d(x_i'', y_i'')$$

$$= \frac{1}{N} \frac{\sum_i \sqrt{x_i'^2 + y_i'^2}\, \sqrt{2}}{C'} = \frac{\sqrt{2}\, C'}{C'} = \sqrt{2}.$$

$$\therefore \begin{bmatrix} x_i'' \\ y_i'' \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} \sqrt{2}/C' & 0 & 0 \\ 0 & \sqrt{2}/C' & 0 \\ 0 & & 1 \end{bmatrix}}_{T_{21}} \begin{bmatrix} x_i' \\ y_i' \\ 1 \end{bmatrix}$$

$$\therefore \begin{bmatrix} x_i'' \\ y_i'' \\ 1 \end{bmatrix} = T_{21} T_{11} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

$$T_1 = T_{21}T_{11}$$

After transformation, we have

in image ① $\leftarrow$ $\boxed{P_1' = T_1 P_1}$; $\boxed{P_2' = T_2 P_2}$ $\rightarrow$ in image ②

$$P_2 = H P_1 \Rightarrow T_2^{-1} P_2' = H\, T_1^{-1} P_1'$$

$$\Rightarrow P_2' = \underbrace{T_2 H T_1^{-1}}_{H'} P_1'$$

Find the homography H' using method-1; then compute

$$H' = T_2 H T_1^{-1}$$

$$\boxed{T_2^{-1} H' T_1 = H}$$

* Using RANSAC for robustness:

Minimum no. of points required : 4 for obtaining one homography.

Do 'N' trials, 'p' inlier probability.

Take randomly 4 feature matches out of N_0 possible matches in each trial.

Compute homography.

$[P]_\times$: cross product form of a vector.

$$p' = Hp; \quad \underbrace{[p']_\times Hp}= 0$$

$$\boxed{\begin{matrix}|a_i^T h| < \varepsilon \\ \& \\ |b_i^T h| < \varepsilon\end{matrix}} \quad \text{(or)} \quad \boxed{\left|[p']_\times Hp\right| < \varepsilon}$$

a_i^T, b_i^T : rows corresponding to one match in 'A' in (1).

ε - tolerance

classify other points as inliers/outliers according to the above relations.

choose that H_{max} which has max. inliers

→ then recompute $\bar{H}$ using all those inliers.

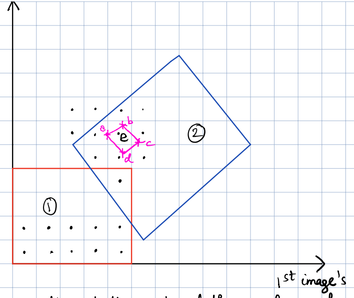
$$\underset{2N \times 9}{A}\,\bar{h} = 0$$

N: max number of inliers for H_{max}.

→ The trials are meant to give us all the inliers, once we get them, we want to fit homography using all of them.

Interpolation:

→ Want to bring all the images into one frame of reference (say the first image's frame of reference).



Need intensities at the centre of the pixels;

1 Bring the pixel locations of other images into the 1st image frame of reference, using the homography transformations.

$$P_2 = H_{12} P_1$$

Use H_{12} to bring points in 2nd image to 1st image.

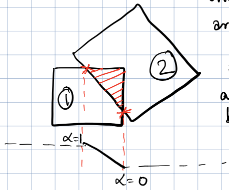
$$H_{12}^{-1} P_2 = P_1$$

Those are no longer integer locations.

2 Then interpolate using for example Bicubic

interpolation (using the 4 nearest neighbours, in the above figure, intensity at 'e' is computed using those at 'a', 'b', 'c', 'd' :→ which are fractional locations obtained after step ①.)

BLENDING



Once all the images are in the same frame of reference, intensities in the overlap regions are computed using blending.

① Feathering
α varies linearly from 1 to 0 in the overlap region (as discussed in class)

$$I = \alpha I_1 + (1-\alpha) I_2$$

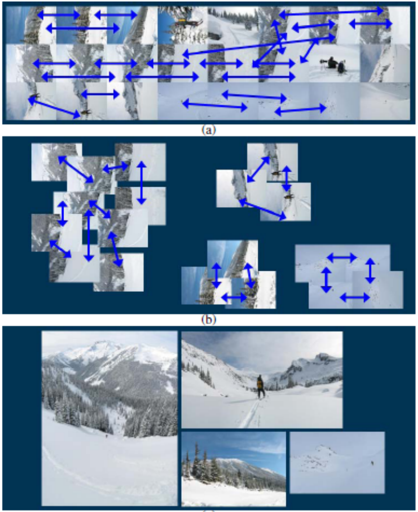
② Using Image Pyramids (Not discussed in class, one could look it up.

Note: Interested people can also use M-estimators for robustness which wasn't discussed in the class.

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x_2 & -y_1x_2 & -x_2 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y_2 & -y_1y_2 & -y_2 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ \vdots \\ h_{33} \end{bmatrix} = \mathbf{0}$$

Projective Scaling?

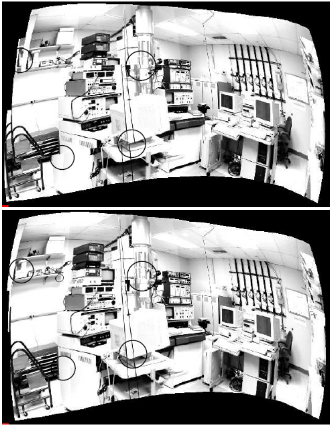
- Are all RHS zeros the same?
- What happens if we set $h_{33} = 1$?
- Yields an $\boldsymbol{Ah} = \boldsymbol{b}$ problem

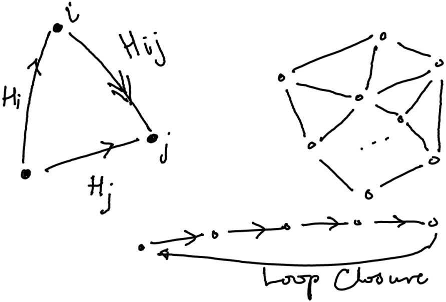


(a) (b) (c)

Figure 9.11 Recognizing panoramas (Brown, Szeliski, and Winder 2005), figures courtesy of Matthew Brown: (a) input images with pairwise matches; (b) images grouped into connected components (panoramas); (c) individual panoramas registered and blended into stitched composites.

Recognising Panoramas





Consistency

- Pairwise alignment causes drift
- Use all relationships ("loop closures")
- $\boldsymbol{H}_{ij} = \boldsymbol{H}_j \boldsymbol{H}_i^{-1}$
- Significantly reduces inconsistencies
- Well-developed method of motion averaging