GENERAL

- Grace Hopper Attendees
  - We’re trying to get this week’s lectures recorded.
  - You can submit problem set 2 by next Friday 11:59 PM, without accruing late days.
  - But you must submit by Friday! (We may not grade problem sets submitted after that)

LAST TIME: TRANSFORMATIONS

- We talked about translation, rotation, euclidean, ... affine transformations as linear operations on homogeneous co-ordinates.
- Upto affine, parallel lines stay parallel.
- But the most general form, is where the linear transform has no restrictions. Called a homography.

HOMOGENEOUS CO-ORDINATES: 2D

Most general form:

\[ p' = Hp \]

where \( H \) is a general invertible \( 3 \times 3 \) matrix.

- Called a projective transform or homography.
- All bets are off! Parallel lines can now intersect. Maps quadrilaterals to quadrilaterals.
- The only certainty you have is that straight lines will remain straight (won’t map straight lines to curves).

Why?

\[ p = \alpha p_1 + (1 - \alpha)p_2 \Rightarrow p' = Hp = aH p_1 + (1 - a)Hp_2 = \alpha' p_1 + (1 - \alpha)p_2' \]
HOMOGENEOUS CO-ORDINATES: 2D

Most general form:

\[ p' = Hp \]

where \( H \) is a general invertible \( 3 \times 3 \) matrix.

- Called a projective transform or **homography**.
- All bets are off! Parallel lines can now intersect. Maps quadrilaterals to quadrilaterals.
- The only certainty you have is that straight lines will remain straight (won’t map straight lines to curves).
- \( H \) is a \( 3 \times 3 \) matrix defined upto scale. So 8 degrees of freedom.
- **Hierarchy of Transforms**
  - Translation (2 dof) < Euclidean (3 dof) < Affine (6 dof) < Homography (8 dof)
- Defines mapping of co-ordinates of corresponding points in two images taken from different views:
  - If all corresponding points lie on a plane in the world.
  - If only the camera orientation has changed in two views (camera center is at the same place)

ESTIMATION

I know a bunch of pairs of points \( (p'_i, p_i) \), and want to find \( H \) such that:

\[ p'_i \sim Hp_i, \ \forall i \]

- How many unknowns? 8 (defined upto scale)
- How many equations for four points? 8 (2 x 4)

But how do we write these equations for equality upto scale?

Realize that if \( p_1 \sim p_2 \), then \( p_1 \) and \( p_2 \) are scaled versions of each other. The **angle** between them is 0.

- Again, thinking back to the vector cross product:
  \[ u \times v = ||u|| ||v|| \sin \theta \hat{n}, \ \text{where} \ \hat{n}^T u = \hat{n}^T v = 0 \]
  \[ p'_i \times (Hp_i) = 0 \]

Recall: \( u \times v = [(u_2v_3 - u_3v_2), (u_3v_1 - u_1v_3), (u_1v_2 - u_2v_1)]^T \)

ESTIMATION

How to re-write equations as linear systems in elements of \( H \)

\[ H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \]

\[ \Rightarrow h = [h_{11}, h_{12}, h_{13}, h_{21}, h_{22}, h_{23}, h_{31}, h_{32}, h_{33}]^T \]

\[ p'_i \times (Hp_i) = 0 \]

\[ \Rightarrow A_i h = 0 \]

Let \( p_i = [p_{ix}, p_{iy}, p_{iz}]^T \):

\[ H p_i = \begin{bmatrix} h_{11} p_{ix} + h_{12} p_{iy} + h_{13} p_{iz} \\ h_{21} p_{ix} + h_{22} p_{iy} + h_{23} p_{iz} \\ h_{31} p_{ix} + h_{32} p_{iy} + h_{33} p_{iz} \end{bmatrix} = \begin{bmatrix} p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & p_{ix} & p_{iy} & p_{iz} \end{bmatrix} h \]

Cross product of two vectors as Matrix vector Multiply

\[ u \times v = \begin{bmatrix} u_y v_z - u_z v_y \\ u_z v_x - u_x v_z \\ u_x v_y - u_y v_x \end{bmatrix} = \begin{bmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{bmatrix} v \]
So we have $Ah = 0$ and want to find $h$ up to scale. $A$ has rank $2n$ and $h$ has 9 elements.

Case 1: $n = 4$ non-collinear points.

- Trivial solution is $h = 0$. But want to avoid this.
- Cast as finding $Ah = 0$ such that $\|h\| = 1$.
- Since $A$ is exactly rank 8, there exists such a solution and it is unique (up to sign).
- Can find using eigen-decomposition / SVD.
- $A = UDV^T$ where $D$ is diagonal with last element 0. $h$ is the last column of $V$.

Case 2: $n > 4$ non-collinear points.

- Over-determined case. Want to find “best” solution.
- $h = \arg\min_h \|Ah\|^2$, $\|h\| = 1$
- Same solution, except that instead of taking 0 singular value, we take minimum singular value.
- $\|Ah\|^2 = (Ah)^T(Ah) = h^T(A^T A)h$
- Minimized by unit vector corresponding to lowest eigenvalue of $A^T A$, or lowest singular value of $A$. 

The cross product gives us 3 equations, so $A_j$ is $3 \times 9$.

But, one of the rows of $A_j$ is a linear combination of the other ($A_j$ has rank 2). Can choose to keep only two rows, or all three.

Stacking all the $A_j$ matrices for all different correspondences, we get:

$Ah = 0$

$A$ is $2n \times 9$ or $3n \times 9$ matrix, where $n$ is number of correspondences. Rank($A$) is at most $2n$.

Rank exactly equal to $2n$ if no three points are collinear.
Other approaches:

- Instead of measuring $\|AH\|^2$, might want to measure explicit geometric distance.
- Minimize distance in mapped cartesian co-ordinates (re-projection error).
- Involves division, no longer linear in $H$. Iterative methods.
- See "Multiple View Geometry in Computer Vision," Hartley & Zisserman: Section 4.2 (or really, the whole book for a thorough discussion of geometry)

Degeneracy constraints:
- No $l$ vector a linear combination of other two
- No more than two parallel lines
- No more than two lines intersecting at same point.

Not really surprising: can always find intersections of lines and match those.

Application:
- Blending / Compositing Images

Given a set of Correspondences:

Option 1
- Estimate Homography from Source to Destination
- For every pixel in the input, apply homography to determine position in the output, and copy the pixel there.

Round pixel location
Leaves gaps if there's scaling.
If multiple pixels map to same place, should average.
For a bunch of samples denoted by set $C$:

$$h = \arg \min_h \sum_{i \in C} E_i(h), \quad \text{for some error function } E \text{ from sample } i$$

Robust Version:

$$h = \arg \min_h \sum_{i \in C} \min(e, E_i(h))$$

- Limits the extent to which an erroneous sample can hurt
- If a specific $E_i > \epsilon$, what is its gradient with respect to $h$?

So if I knew which $i \in C$ would have $E_i > \epsilon$ for the correct $h$,

$$h = \arg \min_{h} \sum_{i : E_i \leq \epsilon} E_i(h)$$

Drop those samples, and solve the normal way (SVD, etc.)
ROBUST FITTING

Iterative Version:

\[ h = \arg \min_h \sum_{i \in C} \min(c, E_i(h)) \]

1. Fit the best \( h \) to all samples in full set \( C \).
2. Given the current estimate of \( h \), compute the inlier set \( C' = \{ i : E_i(h) \leq \epsilon \} \)
3. Update estimate of \( h \) by minimizing error over only the inlier set \( C' \)
4. Goto step 2

Will this converge? Yes. Cost will never increase.

Stop when it stops decreasing. (Might oscillate between two solutions with same cost).

So method converges to some solution. Is it the global minimum?

No. It’s possible that if I made one more point an outlier, that would increase its error to \( > \epsilon \), but reduce other errors by a lot.

Fundamentally a combinatorial problem. Only way to solve exactly is to consider all possible sub-sets of \( C \) as outlier sets.