CSE 559A: Computer Vision

Fall 2020: T-R: 11:30-12:50pm @ Wrighton 300 / Zoom

Instructor: Ayan Chakrabarti (ayan@wustl.edu).
Course Staff: Adith Boloor, Patrick Williams

http://www.cse.wustl.edu/~ayan/courses/cse559a/

Oct 22, 2020
GENERAL

- Last hybrid class. Online only from next week.
  - Office hours (separate Zoom link) with me Tu, Th 1-1:30pm.
- Recitation tomorrow for PSET 2 (will be recorded).
- PSET 2 due Tuesday.
• **Option 1**: Read and analyze a computer vision paper.
  - Recent or classic from ICCV, ECCV, CVPR. (Suggestions posted)
  - Either implement, or if implementation available, modify / analyze.
  - Key is to demonstrate you understood method, and why it was needed.

• **Option 2**: Apply what you’ve learned in class to a problem you care about.
  - Read up on most relevant related work.
  - Implement adapted method for your problem.
  - Analyze results. Did it work? If so, how well. If not, why not.
    - (Should not involve work that is submitted to another course for credit.)

• Should be roughly 2x the effort of a problem set. (Let’s say, problem set 2)
• Avoid anything that requires training neural networks!
PROJECT

Read through the project section on the course website

Grading

• 25 points Report
  ▪ Abstract: 3 pts (One paragraph succinct summary)
  ▪ Introduction/Motivation: 5 pts
    Why is this problem important, what is the vision task, prelude to rest of the report.
  ▪ Related work: 4 pts
    How have other people solved it? What are other similar problems? Read, describe.
  ▪ Description / Experiments / Technical Correctness: 10 pts
  ▪ Conclusion: 3 pts

2-3 Paragraph Proposal is due 11:59pm November 3rd.

Submit as a text file through git (repo will be created shortly).
LAST TIME

- Discussed Homogeneous Co-ordinates
- Book-keeping trick that makes non-linear projection linear.
- Talks about lines and points in 2D in homogeneous co-ordinates.
- Talked about expressing various 2D transformations as matrix multiplications with homogeneous co-ordinates.
- Homographies are the most general form of transformations:
  - Any 3x3 invertible matrix (defined upto scale)
  - Doesn’t preserve area, length, angles, orientation.
  - Straight lines stay straight: but can make parallel lines intersect.
- The transformation between two images of the same scene taken by different cameras is a homography if:
  - Either, the scene (or the part of the scene you’re applying this to) is entirely planar.
  - Or, the two cameras had the same camera center, and only differed in viewing direction.
I know a bunch of pairs of points \((p'_i, p_i)\), and want to find \(H\) such that:

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

Say I knew the length and width of the rug.

Estimate \(H\), apply to all points, measure lengths in meters!
I know a bunch of pairs of points $(p'_i, p_i)$, and want to find $H$ such that:

$$p'_i \sim Hp_i, \quad \forall i$$

- How many unknowns?
- How many equations for four points?
I know a bunch of pairs of points \((p'_i, p_i)\), and want to find \(H\) such that:

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

- How many unknowns? 8 (defined up to scale)
- How many equations for four points? 8 \((2 \times 4)\)

But how do we write these equations for equality up to 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:

\[ u \times v = \|u\|\|v\| \sin \theta \hat{n}, \quad \text{where} \quad \hat{n}^T u = \hat{n}^T v = 0 \]

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

Recall: \(u \times v = [ (u_2 v_3 - u_3 v_2), (u_3 v_1 - u_1 v_3), (u_1 v_2 - u_2 v_1) ]^T\)
How to re-write equations as linear systems in elements of $H$
How to re-write equations as linear systems in elements of $H$

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \quad \Rightarrow \quad h = [h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8, h_9]^T$$
How to re-write equations as linear systems in elements of $H$

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \quad \Rightarrow \quad h = [h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8, h_9]^T$$

$$p'_i \times (Hp_i) = 0 \quad \Rightarrow \quad A_i h = 0$$
How to re-write equations as linear systems in elements of $H$

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \implies h = [h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8, h_9]^T$$

$$p_i' \times (Hp_i) = 0 \implies A_i h = 0$$

Let $p_i = [p_{ix}, p_{iy}, p_{iz}]^T$.
How to re-write equations as linear systems in elements of $H$

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \quad \Rightarrow \quad h = [h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8, h_9]^T$$

$$p_i' \times (H p_i) = 0 \quad \Rightarrow \quad A_i h = 0$$

Let $p_i = [p_{ix}, p_{iy}, p_{iz}]^T$:

$$H p_i = \begin{bmatrix} h_1 p_{ix} + h_2 p_{iy} + h_3 p_{iz} \\ h_4 p_{ix} + h_5 p_{iy} + h_6 p_{iz} \\ h_7 p_{ix} + h_8 p_{iy} + h_9 p_{iz} \end{bmatrix}$$
How to re-write equations as linear systems in elements of $H$

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \quad \Rightarrow \quad h = [h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8, h_9]^T$$

$$p_i' \times (Hp_i) = 0 \quad \Rightarrow \quad A_i h = 0$$

Let $p_i = [p_{ix}, p_{iy}, p_{iz}]^T$:

$$Hp_i = \begin{bmatrix} h_1 p_{ix} + h_2 p_{iy} + h_3 p_{iz} \\ h_4 p_{ix} + h_5 p_{iy} + h_6 p_{iz} \\ h_7 p_{ix} + h_8 p_{iy} + h_9 p_{iz} \end{bmatrix} = \begin{bmatrix} \vdots \end{bmatrix}$$

?? $x$?
How to re-write equations as linear systems in elements of $H$

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \quad \Rightarrow \quad h = [h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8, h_9]^T$$

$$p_i' \times (Hp_i) = 0 \quad \Rightarrow \quad A_i h = 0$$

Let $p_i = [p_{ix}, p_{iy}, p_{iz}]^T$:

$$H p_i = \begin{bmatrix} h_{1p_{ix}} + h_{2p_{iy}} + h_{3p_{iz}} \\ h_{4p_{ix}} + h_{5p_{iy}} + h_{6p_{iz}} \\ h_{7p_{ix}} + h_{8p_{iy}} + h_{9p_{iz}} \end{bmatrix} = \begin{bmatrix} \text{green} \\ \text{red} \\ \text{red} \end{bmatrix} h$$

$3 \times 9$
ESTIMATION

How to re-write equations as linear systems in elements of $H$

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \quad \Rightarrow \quad h = [h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8, h_9]^T$$

$$p'_i \times (Hp_i) = 0 \quad \Rightarrow \quad A_i h = 0$$

Let $p_i = [p_{ix}, p_{iy}, p_{iz}]^T$:

$$H p_i = \begin{bmatrix} h_1 p_{ix} + h_2 p_{iy} + h_3 p_{iz} \\ h_4 p_{ix} + h_5 p_{iy} + h_6 p_{iz} \\ h_7 p_{ix} + h_8 p_{iy} + h_9 p_{iz} \end{bmatrix} = \begin{bmatrix} p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} h \end{bmatrix}$$

$$3 \times 9$$
How to re-write equations as linear systems in elements of $H$

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \quad \Rightarrow \quad h = [h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8, h_9]^T$$

$$p_i' \times (Hp_i) = 0 \quad \Rightarrow \quad A_i h = 0$$

Let $p_i = [p_{ix}, p_{iy}, p_{iz}]^T$:

$$Hp_i = \begin{bmatrix} h_1 p_{ix} + h_2 p_{iy} + h_3 p_{iz} \\ h_4 p_{ix} + h_5 p_{iy} + h_6 p_{iz} \\ h_7 p_{ix} + h_8 p_{iy} + h_9 p_{iz} \end{bmatrix} = \begin{bmatrix} p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 \end{bmatrix} h$$

$$3 \times 9$$
How to re-write equations as linear systems in elements of H

\[
H = \begin{bmatrix}
  h_1 & h_2 & h_3 \\
  h_4 & h_5 & h_6 \\
  h_7 & h_8 & h_9 \\
\end{bmatrix}
\Rightarrow h = [h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8, h_9]^T
\]

\[
p'_i \times (Hp_i) = 0 \quad \Rightarrow \quad A_i h = 0
\]

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

\[
Hp_i = \begin{bmatrix}
  h_1 p_{ix} + h_2 p_{iy} + h_3 p_{iz} \\
  h_4 p_{ix} + h_5 p_{iy} + h_6 p_{iz} \\
  h_7 p_{ix} + h_8 p_{iy} + h_9 p_{iz} \\
\end{bmatrix} = \begin{bmatrix}
  p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & p_{ix} & p_{iy} & p_{iz} \\
\end{bmatrix} h
\]
How to re-write equations as linear systems in elements of $H$

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \quad \Rightarrow \quad h = [h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8, h_9]^T$$

$$p'_i \times (Hp_i) = 0 \quad \Rightarrow \quad A_i h = 0$$

Let $p_i = [p_{ix}, p_{iy}, p_{iz}]^T$:

$$Hp_i = \begin{bmatrix} h_1 p_{ix} + h_2 p_{iy} + h_3 p_{iz} \\ h_4 p_{ix} + h_5 p_{iy} + h_6 p_{iz} \\ h_7 p_{ix} + h_8 p_{iy} + h_9 p_{iz} \end{bmatrix} = \begin{bmatrix} p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & p_{ix} & p_{iy} & p_{iz} \end{bmatrix} h$$

Cross product of two vectors as Matrix vector Multiply

$$u \times v = \begin{bmatrix} u_y v_z - u_z v_y \\ u_z v_x - u_x v_z \\ u_x v_y - u_y v_x \end{bmatrix}$$
ESTIMATION

How to re-write equations as linear systems in elements of $H$

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \quad \Rightarrow \quad h = [h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8, h_9]^T$$

$$p'_i \times (Hp_i) = 0 \quad \Rightarrow \quad A_i h = 0$$

Let $p_i = [p_{ix}, p_{iy}, p_{iz}]^T$:

$$Hp_i = \begin{bmatrix} h_1 p_{ix} + h_2 p_{iy} + h_3 p_{iz} \\ h_4 p_{ix} + h_5 p_{iy} + h_6 p_{iz} \\ h_7 p_{ix} + h_8 p_{iy} + h_9 p_{iz} \end{bmatrix} = \begin{bmatrix} p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & p_{ix} & p_{iy} & p_{iz} \end{bmatrix} h$$

Cross product of two vectors as Matrix vector Multiply

$$u \times v = \begin{bmatrix} u_y v_z - u_z v_y \\ u_z v_x - u_x v_z \\ u_x v_y - u_y v_x \end{bmatrix} = \begin{bmatrix} \text{3x3} \end{bmatrix} v$$
ESTIMATION

How to re-write equations as linear systems in elements of $H$

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \quad \Rightarrow \quad h = [h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8, h_9]^T$$

$$p_i' \times (Hp_i) = 0 \quad \Rightarrow \quad A_i h = 0$$

Let $p_i = [p_{ix}, p_{iy}, p_{iz}]^T$:

$$Hp_i = \begin{bmatrix} h_1 p_{ix} + h_2 p_{iy} + h_3 p_{iz} \\ h_4 p_{ix} + h_5 p_{iy} + h_6 p_{iz} \\ h_7 p_{ix} + h_8 p_{iy} + h_9 p_{iz} \end{bmatrix} = \begin{bmatrix} p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & p_{ix} & p_{iy} & p_{iz} \end{bmatrix} h$$

Cross product of two vectors as Matrix vector Multiply

$$u \times v = \begin{bmatrix} u_y v_z - u_z v_y \\ u_z v_x - u_x v_z \\ u_x v_y - u_y v_x \end{bmatrix} = \begin{bmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{bmatrix} v$$
How to re-write equations as linear systems in elements of $H$

\[
H = \begin{bmatrix}
    h_1 & h_2 & h_3 \\
    h_4 & h_5 & h_6 \\
    h_7 & h_8 & h_9 
\end{bmatrix} \quad \Rightarrow \quad h = [h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8, h_9]^T
\]

\[
p_i' \times (Hp_i) = 0 \quad \Rightarrow \quad A_i h = 0
\]

Let $p_i = [p_{ix}, p_{iy}, p_{iz}]^T$:

\[
Hp_i = \begin{bmatrix}
    h_1 p_{ix} + h_2 p_{iy} + h_3 p_{iz} \\
    h_4 p_{ix} + h_5 p_{iy} + h_6 p_{iz} \\
    h_7 p_{ix} + h_8 p_{iy} + h_9 p_{iz}
\end{bmatrix} = \begin{bmatrix}
    p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 & 0 & 0 & 0 \\
    0 & 0 & 0 & p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 \\
    0 & 0 & 0 & 0 & 0 & 0 & p_{ix} & p_{iy} & p_{iz}
\end{bmatrix} h
\]

Cross product of two vectors as Matrix vector Multiply

\[
u \times v = \begin{bmatrix}
    u_v v_z - u_z v_y \\
    u_z v_x - u_x v_z \\
    u_x v_y - u_y v_x
\end{bmatrix} = \begin{bmatrix}
    0 & -u_z & u_y \\
    u_z & 0 & -u_x
\end{bmatrix} v
\]
ESTIMATION

How to re-write equations as linear systems in elements of $H$

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \Rightarrow h = [h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8, h_9]^T$$

$$p_i' \times (Hp_i) = 0 \Rightarrow A_i h = 0$$

Let $p_i = [p_{ix}, p_{iy}, p_{iz}]^T$:

$$Hp_i = \begin{bmatrix} h_1p_{ix} + h_2p_{iy} + h_3p_{iz} \\ h_4p_{ix} + h_5p_{iy} + h_6p_{iz} \\ h_7p_{ix} + h_8p_{iy} + h_9p_{iz} \end{bmatrix} = \begin{bmatrix} p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & p_{ix} & p_{iy} & p_{iz} \end{bmatrix} h$$

Cross product of two vectors as Matrix vector Multiply

$$u \times v = \begin{bmatrix} u_yv_z - u_zv_y \\ u_zv_x - u_xv_z \\ u_xv_y - u_yv_x \end{bmatrix} = \begin{bmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{bmatrix} v$$
How to re-write equations as linear systems in elements of $H$

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \quad \Rightarrow \quad h = [h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8, h_9]^T$$

$$p_i' \times (Hp_i) = 0 \quad \Rightarrow \quad A_i h = 0$$

Let $p_i = [p_{ix}, p_{iy}, p_{iz}]^T$:

$$Hp_i = \begin{bmatrix} h_1 p_{ix} + h_2 p_{iy} + h_3 p_{iz} \\ h_4 p_{ix} + h_5 p_{iy} + h_6 p_{iz} \\ h_7 p_{ix} + h_8 p_{iy} + h_9 p_{iz} \end{bmatrix} = \begin{bmatrix} p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & p_{ix} & p_{iy} & p_{iz} \end{bmatrix} h$$

Cross product of two vectors as Matrix vector Multiply

$$u \times v = \begin{bmatrix} u_y v_z - u_z v_y \\ u_z v_x - u_x v_z \\ u_x v_y - u_y v_x \end{bmatrix} = \begin{bmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{bmatrix} v$$

Note that this has rank 2. Why?
ESTIMATION

How to re-write equations as linear systems in elements of $H$

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \Rightarrow h = [h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8, h_9]^T$$

$$p_i' \times (Hp_i) = 0 \Rightarrow A_i h = 0$$

Let $p_i = [p_{ix}, p_{iy}, p_{iz}]^T$:

$$Hp_i = \begin{bmatrix} h_1p_{ix} + h_2p_{iy} + h_3p_{iz} \\ h_4p_{ix} + h_5p_{iy} + h_6p_{iz} \\ h_7p_{ix} + h_8p_{iy} + h_9p_{iz} \end{bmatrix} = \begin{bmatrix} p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & p_{ix} & p_{iy} & p_{iz} \end{bmatrix} h$$

Cross product of two vectors as Matrix vector Multiply

$$u \times v = \begin{bmatrix} u_yv_z - u_zv_y \\ u_zv_x - u_xv_z \\ u_xv_y - u_yv_x \end{bmatrix} = \begin{bmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{bmatrix} v$$

Note that this has rank 2. Why?  
Third row = - $u_y/u_z \times$ First row  
- $u_y/u_z \times$ Second row
\[ p' \times (H_{p_i}) = 0 \Rightarrow A_i h = 0 \]

\[
Hp_i = \begin{bmatrix}
h_1 p_{ix} + h_2 p_{iy} + h_3 p_{iz} \\
h_4 p_{ix} + h_5 p_{iy} + h_6 p_{iz} \\
h_7 p_{ix} + h_8 p_{iy} + h_9 p_{iz}
\end{bmatrix} = \begin{bmatrix}
p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & p_{ix} & p_{iy} & p_{iz}
\end{bmatrix} h
\]

\[ u \times v = \begin{bmatrix}
  u_y v_z - u_z v_y \\
u_z v_x - u_x v_z \\
u_x v_y - u_y v_x
\end{bmatrix} = \begin{bmatrix}
  0 & -u_z & u_y \\
u_z & 0 & -u_x \\
-u_y & u_x & 0
\end{bmatrix} v
\]

\[
p' \times (H_{p_i}) = \begin{bmatrix}
0 & -p'_{iz} & p'_{iy} \\
p'_{ix} & 0 & -p'_{ix} \\
-p'_{iy} & p'_{ix} & 0
\end{bmatrix} \begin{bmatrix}
p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & p_{ix} & p_{iy} & p_{iz} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & p_{ix} & p_{iy} & p_{iz}
\end{bmatrix} h
\]
The cross product gives us 3 equations, so $A_i$ is $3 \times 9$.

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

Stacking all the $A_i$ 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.
So we have $Ah = 0$ and want to find $h$ upto 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 (upto sign).
- Can find using eigen-decomposition / SVD.
  - $A = USV^T$ where $S$ is “diagonal” with last element 0. $h$ is the last column of $V$.
    - $V$ will be $9 \times 9$ (call `np.linalg.svd` with `full_matrices=True`).

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

- Over-determined case. Want to find “best” solution.
So we have $Ah = 0$ and want to find $h$ upto 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 (upto sign).
- Can find using eigen-decomposition / SVD.
  - $A = USV^T$ where $S$ is “diagonal” with last element 0. $h$ is the last column of $V$.
    - $V$ will be $9 \times 9$ (call `np.linalg.svd` with `full_matrices=True`).

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

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

Estimation from Lines

- How does a homography transform a line:

\[ l^T p = 0 \leftrightarrow l'^T p' = 0 \]

\[ l^T H^{-1} Hp = 0 \Rightarrow (H^{-T} l)^T (Hp) = 0 \]

\[ l' = H^{-T} l \Rightarrow l = H^T l' \]

- If we find four pairs of corresponding lines, we can get a similar set of equations for \( l_i = H^T l'_i \) as for points.

- Get equations from \( l_i \times (H^T l'_i) = 0 \) for elements of \( H \).
Matching with point correspondences

\[ p_i' \times H p_i = 0 \]
Matching with line correspondences

\[ l_i \times H^T l'_i = 0 \]
Matching with line correspondences

Homography

\[ l_i \times H^T l_i' = 0 \]

Degeneracy constraints?
Matching with line correspondences

Degeneracy constraints?
- No line vector a linear combination of other two

\[ l_i \times H^T l_i' = 0 \]
Matching with line correspondences

Degeneracy constraints:
- No line vector a linear combination of other two
- No more than two parallel lines

\[ l_i \times H^T l'_i = 0 \]
Matching with line correspondences

Degeneracy constraints?
- No line vector a linear combination of other two
- No more than two parallel lines

\[ l_i \times H^T l'_i = 0 \]

Not really surprising:
can always find intersections of lines and match those.
Degeneracy constraints?
- No 1 vector a linear combination of other two
- No more than two parallel lines
- No more than two lines intersecting at same point.

\[ l_i \times H^T l_i' = 0 \]

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

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)
Application:

- Blending / Compositing Images
Say you want to take part of source, and warp it and place / blend it in destination:
Say you want to take part of source, and warp it and place / blend it in destination:

Given a set of Correspondences:
Say you want to take part of source, and warp it and place / blend it in destination:

Given a set of Correspondences:

**Option 1**
- Estimate Homography from Source to Destination
Say you want to take part of source, and warp it and place / blend it in destination:

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.
Say you want to take part of source, and warp it and place / blend it in destination:

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.
Say you want to take part of source, and warp it and place / blend it in destination:

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.
Say you want to take part of source, and warp it and place / blend it in destination:

![Diagram showing source and destination with a transformation](image)

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.
Say you want to take part of source, and warp it and place / blend it in destination:

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.
SAMPLING

Say you want to take part of source, and warp it and place / blend it in destination:

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.
Say you want to take part of source, and warp it and place / blend it in destination:

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.
Say you want to take part of source, and warp it and place / blend it in destination:

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.
SAMPLING

Say you want to take part of source, and warp it and place / blend it in destination:

Given a set of Correspondences:

**Option 2**
- Estimate Homography from Destination to Source
Say you want to take part of source, and warp it and place / blend it in destination:

Given a set of Correspondences:

**Option 2**
- Estimate Homography from Destination to Source
Say you want to take part of source, and warp it and place / blend it in destination:

Given a set of Correspondences:

**Option 2**
- Estimate Homography from Destination to Source
- For each pixel in destination, compute location in source, and copy from there.
Say you want to take part of source, and warp it and place / blend it in destination:

Given a set of Correspondences:

**Option 2**
- Estimate Homography from Destination to Source
- For each pixel in destination, compute location in source, and copy from there.

**Bi-linear Interpolation:**
\[
[I_{(202.3,210.7)} = 0.7*(0.3*I_{(202,210)} + 0.7*I_{(202,211)}) + 0.3*(0.3*I_{(203,210)} + 0.7*I_{(203,211)})]
\]
Say you want to take part of source, and warp it and place / blend it in destination:

Given a set of Correspondences:

**Option 2**
- Estimate Homography from Destination to Source
- For each pixel in destination, compute location in source, and copy from there.

**Bi-linear Interpolation:**
\[
[202.3, 210.7] = 0.7 \times (0.3 \times [202, 210] + 0.7 \times [202, 211]) + 0.3 \times (0.3 \times [203, 210] + 0.7 \times [203, 211])
\]

If out of bounds, leave at zero / gray.
In reality, correspondences will be noisy:
- Automatic detection isn't perfect
- Neither are user clicks!
In reality, correspondences will be noisy:
- Automatic detection isn't perfect
- Neither are user clicks!
In reality, correspondences will be noisy:
- Automatic detection isn't perfect
- Neither are user clicks!

Small perturbations in correspondences leads to small perturbations in estimate.
But sometimes, you have outliers: one correspondence that is completely wrong.

In reality, correspondences will be noisy:
- Automatic detection isn't perfect
- Neither are user clicks!

Small perturbations in correspondences leads to small perturbations in estimate.
But sometimes, you have outliers: one correspondence that is completely wrong.

In reality, correspondences will be noisy:
- Automatic detection isn't perfect
- Neither are user clicks!
But sometimes, you have outliers: one correspondence that is completely wrong.

In reality, correspondences will be noisy:
- Automatic detection isn't perfect
- Neither are user clicks!

Because our estimator tries to minimize the average error across all correspondences, the estimate is significantly skewed.
ROBUST FITTING

But sometimes, you have outliers: one correspondence that is completely wrong.

In reality, correspondences will be noisy:
- Automatic detection isn't perfect
- Neither are user clicks!

Because our estimator tries to minimize the average error across all correspondences, the estimate is significantly skewed.

One large error is as bad as multiple medium-scale errors
ROBUST FITTING

But sometimes, you have outliers: one correspondence that is completely wrong.
Remember, we only needed 4 points to fit H anyway.

In reality, correspondences will be noisy:
- Automatic detection isn't perfect
- Neither are user clicks!

Because our estimator tries to minimize the average error across all correspondences, the estimate is significantly skewed.

One large error is as bad as multiple medium-scale errors
ROBUST FITTING

But sometimes, you have outliers: one correspondence that is completely wrong.
Remember, we only needed 4 points to fit H anyway.
But need to know which 4.

In reality, correspondences will be noisy:
- Automatic detection isn't perfect
- Neither are user clicks!

Because our estimator tries to minimize the average error across all correspondences, the estimate is significantly skewed.

One large error is as bad as multiple medium-scale errors
ROBUST FITTING

For a bunch of samples denoted by set $C$:

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

Robust Version:

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

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

For a bunch of samples denoted by set $C$:

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

Robust Version:

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

- Limits the extent to which an erroneous sample can hurt
- If a specific $E_i > \epsilon$, what is its gradient with respect to $h$? 0

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

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

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

For a bunch of samples denoted by set $C$:

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

Robust Version:

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

- Limits the extent to which an erroneous sample can hurt
- If a specific $E_i > \epsilon$, what is its gradient with respect to $h \neq 0$

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:

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

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?

Consider the original robust cost \( \min(\epsilon, E_i(h)) \). Can step 3 ever increase the cost?
ROBUST FITTING

Iterative Version:

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

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. Go to step 2

Will this converge?

Consider the original robust cost \( \min(\epsilon, E_i(h)) \). Can step 3 ever increase the cost?

- Before step 3, \( h \) had some cost over the inlier set, and cost over each outlier sample was \( > \epsilon \).
- Step 3 finds the value of \( h \) with minimum cost over inlier set. So error can only decrease over inliner set.
- Step 3 can increase or decrease error over outlier set. But increased error doesn’t hurt us, since it was already \( > \epsilon \) before step 3.
ROBUST FITTING

Iterative Version:

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

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.