Lecture 2 - Historical Vision Problems

- Images as points
- Images as functions
- Convolution
  - Filtering
  - Blurring
  - Edge Detection
  - Template matching

$\Big\}$ Tools

Categories:

Reduction	Matching	Fitting
- $X^* = \arg\min_x f(I, x)$	- $X^* = \arg\min_x d(X, X_I)$	- $\Theta^* = \arg\min_\Theta g(I; \Theta)$
- Indexing - Id-free expression recognition - Featurization	- Img search - Optical flow - Img registration	- Model fitting - Neural nets - Structure from motion

→ Why categorize?

- Similar problems = similar solutions

Plan for lecture:

- Work through example problem: image registration
- Learn about structure tensor
- Learn about homographies

Image Registration


Uses:

- Image stitching →
- Camera stabilization
- Motion reconstruction

Goal:

- Find a transform $T(\cdot)$ that maps points in one image to the same real-world points in another.

How?

Idea 1: Direct fitting

- Parametrize T by θ: $T = \begin{bmatrix} \theta_1 & \theta_2 & \theta_3 \\ \theta_4 & \theta_5 & \theta_6 \\ \theta_7 & \theta_8 & 1 \end{bmatrix}$

- Define a loss: $f(I_1, I_2; \theta) = \underbrace{\sum_{(x,y) \in I_1 \cap I_2} \left\| I_1(x,y) - T(I_2(x,y)) \right\|_2^2}$

- Find the θ^* that minimizes f!

→ Intractable ☹

Idea #2: Point fitting

- Identify "unique" or distinctive points in both images
- Match up which ones correspond to the same points
- Fit the transform that maps these points to each other

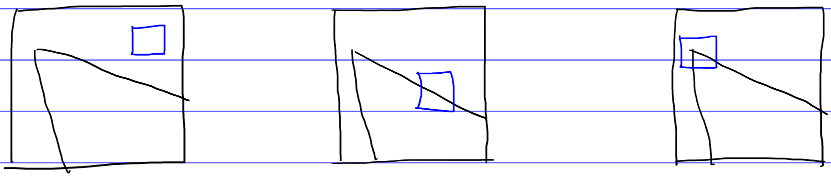
☆ Useful example of:
- Reduction – of points to descriptive features
- Matching – those points to each other
- Fitting – the transform

Step 1: Finding Points of Interest

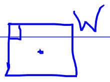
- What makes a point distinct?

★ We want corners. Flat areas and long edges aren't unique enough.

Property of corners: Small shifts in any direction change the neighborhood.



Define a measure of the corner-ness of the neighborhood around a point by how much the neighborhood changes with a small shift $[\Delta x, \Delta y]^T$:



$$f(\Delta x, \Delta y) = \sum_{x,y \in W} \left[I(x,y) - \underline{I(x+\Delta x, y+\Delta y)} \right]^2$$

Taylor Expansion: $I_x = \frac{\partial I}{\partial x}$ $\quad I_y = \frac{\partial I}{\partial y}(x,y)$

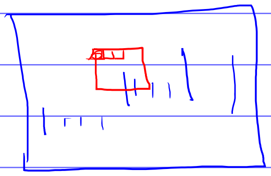
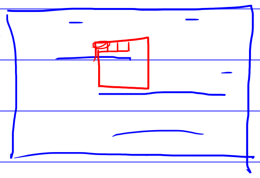
$$I(x+\Delta x, y+\Delta y) \approx I(x,y) + \Delta x \cdot I_x + \Delta y \cdot I_y + \cancel{H.O.T.}$$

$$f(\Delta x, \Delta y) = \sum_{x,y \in W} \left[\cancel{I(x,y)} - \cancel{I(x,y)} + \Delta x I_x + \Delta y I_y \right]^2$$

$$= \sum_{x,y \in W} \left[\Delta x^2 I_x^2 + 2 \Delta x \Delta y I_x I_y + \Delta y^2 I_y^2 \right]$$

$$= \sum_{x,y \in W} \begin{bmatrix} \Delta x & \Delta y \end{bmatrix} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

$$f(\Delta x, \Delta y) = \begin{bmatrix} \Delta x & \Delta y \end{bmatrix} \underbrace{\begin{bmatrix} \sum_{x,y \in W} I_x^2 & \sum_{x,y \in W} I_x I_y \\ \sum_{x,y \in W} I_x I_y & \sum_{x,y \in W} I_y^2 \end{bmatrix}}_{M_W} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

$I_x(x,y) =$  $\quad I_y(x,y) =$ 

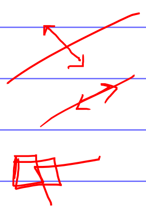
M_W = the structure tensor

Flat : Any $(\Delta x, \Delta y) \to$ low f

Edges : One $(\Delta x, \Delta y) \to$ high f

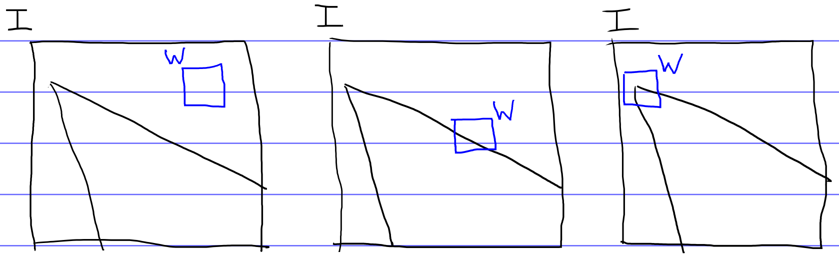
One $(\Delta x, \Delta y) \to$ low f

Corners : Any $(\Delta x, \Delta y) \to$ high f



$$f(\Delta x,\Delta y)=\begin{bmatrix}\Delta x\\ \Delta y\end{bmatrix}^T \underbrace{\begin{bmatrix}\sum_{x,y\in W} I_x^2 & \sum_{x,y\in W} I_x I_y\\ \sum_{x,y\in W} I_x I_y & \sum_{x,y\in W} I_y^2\end{bmatrix}}_{M_w}\begin{bmatrix}\Delta x\\ \Delta y\end{bmatrix}$$

	flat	edges	corner
λ_+	low	high	high
λ_-	low	low	high

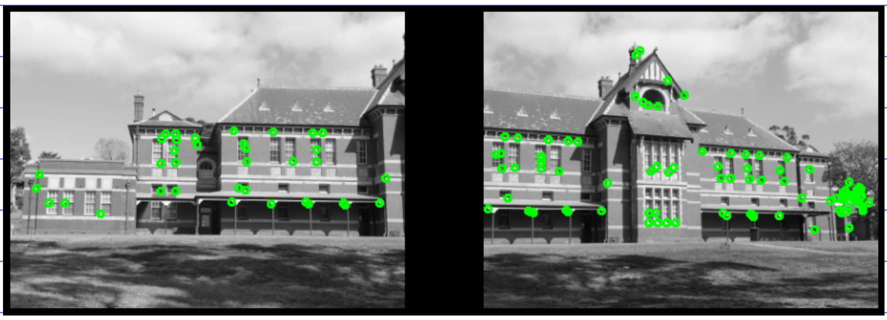


$\lambda_+ \downarrow$	$\lambda_+ \uparrow$	$\lambda_+ \uparrow$
$\lambda_- \downarrow$	$\lambda_- \downarrow$	$\lambda_- \uparrow$

$$\underline{g(W)} = \underset{\Delta x,\Delta y}{\min} f_W(\Delta x,\Delta y) = \underline{\lambda_-} \text{ of } M_W$$

Map of g(W) over both images:

Local maxima:



Next: How do we compare points in the two images to see which ones match each other?

Have: $g(W) : (m \times m) \longmapsto 1D$

↳ good for detection

↳ bad for description

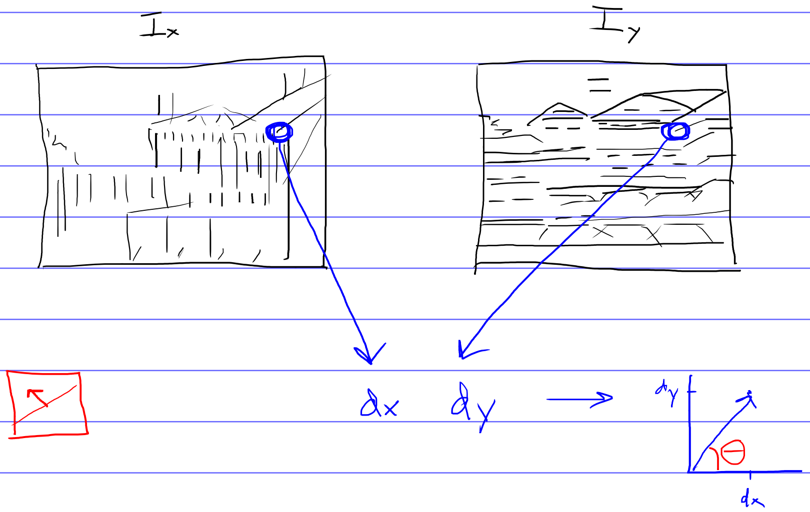
Want: $h(W) : (m \times n) \longmapsto ND$

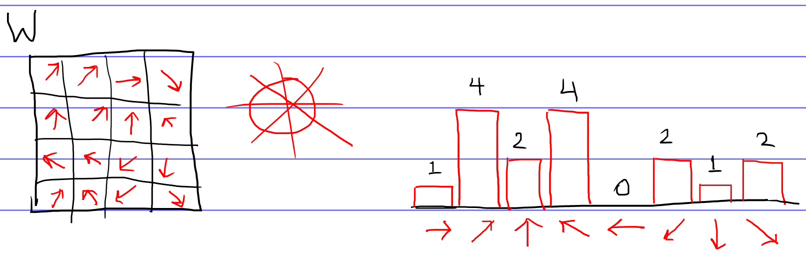
Featurization

- A reduction process where we convert/compress visual information into a uniform representation for comparison or processing

  → Representation is usually a vector, but sometimes a graph, string, or other object

- Simple example algorithm: HOG (Histogram of Oriented Gradients)



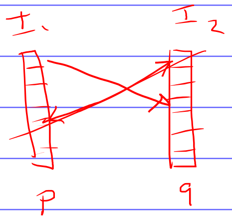


$$\text{feature} = [1,4,2,4,0,2,1,2]/16$$

Next: Which features from image 1 match with which features from image 2?

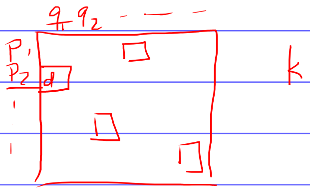
Options:

- Brute force search



- Hungarian Algorithm

$d(p_i, q_i) \geq 0$

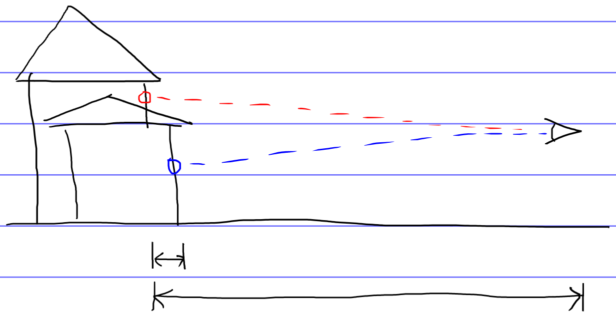


Last step! Find the transform that maps matched points onto eachother.

We have, for $i = \{1, 2, \ldots, n\}$,

$$[x_1^i, y_1^i] \longleftrightarrow [x_2^i, y_2^i]$$

Perspective changes relate coplanar points with a homography.



Homography:

$$w \cdot \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix} \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix}$$

↳ homogeneous coordinates

$$w = gx_2 + hy_2 + 1$$

$$\begin{bmatrix} x_1(gx_2 + hy_2 + 1) \\ y_1(gx_2 + hy_2 + 1) \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix} \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} x_2 a + y_2 \cdot b + c + 0d + 0e + 0 \cdot f - x_1 x_2 g - x_1 y_2 h \\ 0a + 0 \cdot b + 0c + x_2 \cdot d + y_2 e + f - y_1 x_2 g - y_1 y_2 h \end{bmatrix}$$

$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} x_2 & y_2 & 1 & 0 & 0 & 0 & -x_1 x_2 & -x_1 y_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -y_1 x_2 & -y_1 y_2 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \end{bmatrix}$$

$$E_i = \left\| \underbrace{\begin{bmatrix} x_2^i & y_2^i & 1 & 0 & 0 & 0 & -x_1^i x_2^i & -x_1^i y_2^i \\ 0 & 0 & 0 & x_2^i & y_2^i & 1 & -y_1^i x_2^i & -y_1^i y_2^i \end{bmatrix}}_{A_i} H - \underbrace{\begin{bmatrix} x_1^i \\ y_1^i \end{bmatrix}}_{b_i} \right\|_2^2$$

$$\operatorname*{argmin}_H \sum_{i=1}^{n} E_i$$

$$= \left\| \underbrace{\begin{bmatrix} A_1 \\ A_2 \\ A_3 \\ \vdots \\ A_n \end{bmatrix}}_{A} H - \underbrace{\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{bmatrix}}_{b} \right\|_2^2$$

$$H^* = A^\dagger b$$

Recap:

- Three broad categories of CV problem:

	Reduction	Matching	Fitting
Ex: Image Registration	Corner finding Featurization	Feature matching	Homography estimation

- Structure tensor + corner operator

- Featurization with HOG

- Feature matching with Hungarian Alg.

- Homography estimation via OLS