CSE 252B Lecture 02

Owen Jow | January 09, 2019

A Simple Feature Detection Process

in which we detect the coordinates of corner-like entities to high, sub-pixel accuracy

1. COMPUTE IMAGE GRADIENTS

Use the five-point central difference operator.
Convolution kernels:

$$K_y = \frac{1}{12}\begin{bmatrix} -1 \\ 8 \\ 0 \\ -8 \\ 1 \end{bmatrix} \qquad K_x = \frac{1}{12}\begin{bmatrix} -1 & 8 & 0 & -8 & 1 \end{bmatrix}$$

$$I_y = I \circledast K_y \qquad I_x = I \circledast K_x$$

2. COMPUTE STRUCTURE TENSOR ("GRADIENT MATRIX") AT EVERY LOCATION WHERE WINDOW FITS

$$N(x,y) = \begin{bmatrix} \sum\sum_w I_x^2 & \sum\sum_w I_x I_y \\ \sum\sum_w I_x I_y & \sum\sum_w I_y^2 \end{bmatrix}$$

Note: in order to avoid the magnitudes of these values being dependent on window size, we can <u>average</u> over the windows instead of summing.

3. COMPUTE MINOR EIGENVALUE OF STRUCTURE TENSOR AT EACH LOCATION

$$\lambda_{min} = \frac{Tr(N) - \sqrt{Tr(N)^2 - 4\det(N)}}{2}$$

This value indicates bidirectional texturedness.

4. PERFORM NON-MAXIMUM SUPPRESSION ON THE EIGENVALUE IMAGE

a. First filter with local 2D max filter.

$$I_\lambda \rightarrow \left(\text{MAXIMUM FILTER}\right) \rightarrow I_{max}$$

b. Then suppress (set to 0) wherever the signal value < the local max output.

$$\underset{\text{suppressed image}}{J(x,y)} = \begin{cases} 0 & \text{if } I_\lambda(x,y) < I_{max}(x,y) \\ I_\lambda(x,y) & \text{otherwise} \end{cases}$$

5. USE THE FORSTNER CORNER DETECTOR TO LOCALIZE THE ACTUAL CORNER COORDINATES FROM EACH WINDOW

Only run on windows around points in J which have values above some threshold.

Solve $A\underline{x} = \underline{b}$ for $\underline{x}$:

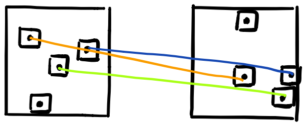
$$\underbrace{\begin{bmatrix} \sum_w\sum I_x^2 & \sum_w\sum I_x I_y \\ \sum_w\sum I_x I_y & \sum_w\sum I_y^2 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x_{corner} \\ y_{corner} \end{bmatrix}}_{\underline{x}} = \underbrace{\begin{bmatrix} \sum_w\sum (x I_x^2 + y I_x I_y) \\ \sum_w\sum (x I_x I_y + y I_y^2) \end{bmatrix}}_{\underline{b}}$$

- use x, y coordinates from the full image
- A is symmetric positive semidefinite (use efficient solver)

Say there are multiple images. We now have a bunch of corners in each one. So that we can do cool stuff later, let's match corners across our (let's say two) images.

A Simple Feature Matching Process

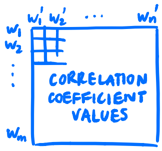
in which we compare windows around each corner to find matches in a one-to-one fashion



1. BUILD CORRELATION COEFFICIENT MATRIX

For each window in image 1, compute the correlation coefficient between that window and each window in image 2. Store all results in matrix.

- window around every corner point in each image
- each correlation coefficient should be $\in [-1, 1]$
  - greater value = better match



m corners / windows in image 1
n corners / windows in image 2

Also create a binary mask matrix of the same shape as the correlation coefficient (CC) matrix, and initialize all values to True.

2. PERFORM ONE-TO-ONE MATCHING

For one-to-one matching, we will use the Russian greedy grandma heuristic (the next-best match should be much worse than the best match).

Repeat until the max value in the masked CC matrix is $\leq$ the similarity threshold:

a. Find the best match, i.e. the element with the maximum value in the masked CC matrix.

b. Store that value, and temporarily replace the element with a -1 in the CC matrix.

c. Find the next-best match according to

$$\max\begin{pmatrix}\text{max value in same row as element,} \\ \text{max value in same col as element}\end{pmatrix}$$

* not a masked operation

d. Set the value of the element back to its original value.

e. if [1 - (best match value)] < [1 - (next-best match value)] * (distance ratio threshold)
   store the feature match
else
   match is not unique enough

f. In the mask, set all values in the best match's row and column to False.
- we're now done considering these corners/windows, whether we found a good match with them or not

Presumably, we now have a set of decent matches.