Aryan Dawer

Department of Computer Science

Dartmouth College

February 5, 2024

PSET 2

For this pset, I shared ideas with my fellow classmate Atul Agarwal, however all the work in this document is my own.

1 Theory Questions

Q2.1.1 How many degrees of freedom does h have?

The vector h has 8 degrees of freedom since it is derived from a 3x3 homography matrix H which is defined up to scale.

Q2.1.2 How many point pairs are required to solve h?

A minimum of 4 point pairs is required to solve for h, since we need 8 points.

Q2.1.3 Derive A_i

Given a pair of corresponding points $\mathbf{x}_1 = (x_1, y_1, 1)^T$ and $\mathbf{x}_2 = (x_2, y_2, 1)^T$, the homography H that maps $\mathbf{x}_1$ to $\mathbf{x}_2$ is represented by the equation $\mathbf{x}_2 \equiv H\mathbf{x}_1$. In terms of their elements, this can be written as:

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

Expanding the multiplication and rearranging the terms, we obtain two equations:

$$x_2 h_{31} x_1 + x_2 h_{32} y_1 + x_2 h_{33} = h_{11} x_1 + h_{12} y_1 + h_{13}$$

$$y_2 h_{31} x_1 + y_2 h_{32} y_1 + y_2 h_{33} = h_{21} x_1 + h_{22} y_1 + h_{23}$$

To form the matrix A_i such that $A_i\mathbf{h} = 0$, where $\mathbf{h}$ is the vectorized form of H (reshaped column-wise), these equations can be rewritten as:

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

From this, we can construct the matrix A_i:

$$A_i = \begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_2x_1 & -x_2y_1 & -x_2 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y_2x_1 & -y_2y_1 & -y_2 \end{bmatrix}$$

Q2.1.4 When solving Ah = 0, in essence you're trying to find the h that exists in the null space of A. What that means is that there would be some non-trivial solution for h such that that product Ah turns out to be 0. What will be a trivial solution for h? Is the matrix A full rank? Why/Why not? What impact will it have on the singular values? What impact will it have on the singular vectors?

The trivial solution to the equation $A\mathbf{h} = \mathbf{0}$ is a zero vector of size 12:

$$\mathbf{h}_{\text{trivial}} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Matrix A is not full rank because a homography is defined by 8 degrees of freedom, not 12. As a result, A has at most rank 8, assuming there are enough constraints from point correspondences.

The singular values of A will reflect its rank deficiency by having at least one singular value that is zero or nearly zero.

The singular vector corresponding to the smallest singular value lies in the null space of A. This vector, when normalized, provides the non-trivial solution to the equation $A\mathbf{h} = \mathbf{0}$ and represents the homography matrix $\mathbf{h}$.

Q3.1 FAST Detector

The FAST (Features from Accelerated Segment Test) detector is different from the Harris corner detector since it uses a circle of 16 pixels around a candidate pixel and classifies the candidate as a corner if there are n pixels which are all brighter than the candidate pixel's intensity plus a threshold or all darker than the candidate's intensity minus a threshold. The Harris detector computes corner response by evaluating the change in intensity for a displacement of the window in each direction. The FAST detector is generally faster than the Harris detector because it does not require the computation of image derivatives or solving eigenvalue problems as in the case of Harris.

Q3.2 BRIEF Descriptor

BRIEF (Binary Robust Independent Elementary Features) descriptors are different from the filter banks discussed in the lectures because BRIEF provides a shortcut to descriptor construction by directly using pixel intensity comparisons to generate a binary string. While filter banks apply a set of filters to an image to capture various features at different scales and orientations, BRIEF simplifies this by considering the results of simple binary tests between pixels in a smoothed image patch. While you could use the Gabor filter to create a GIST descriptor, BRIEF is much more computationally efficient and less memory-intensive, as it generates a binary string rather than a high-dimensional vector of filter responses.

Q3.3 Matching Methods

The Hamming distance between two binary strings is the number of positions at which the corresponding bits are different. It can be computed extremely fast using bitwise operations like XOR between two strings followed by a bit count. The nearest neighbor method involves computing distances from the target point to all other points and selecting the pair with the smallest distance. Euclidean distance is a common metric for this purpose, which is computationally more intensive than the Hamming distance.

Q3.4 Feature Matching

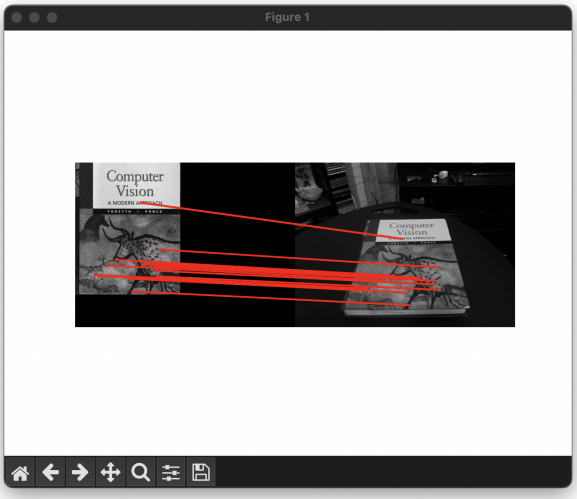

Figure 1: Feature Matching with Sigma=0.15 and ratio=0.67

Q3.5 BRIEF and Rotations

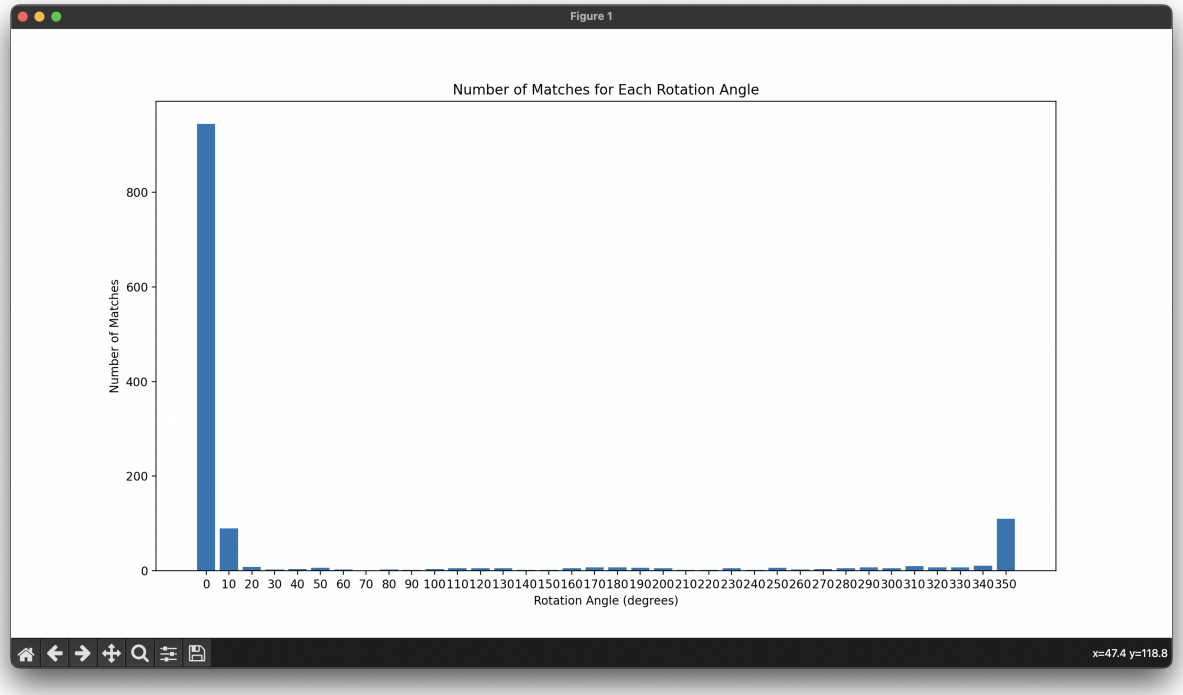

Figure 2: Matches at all orientations

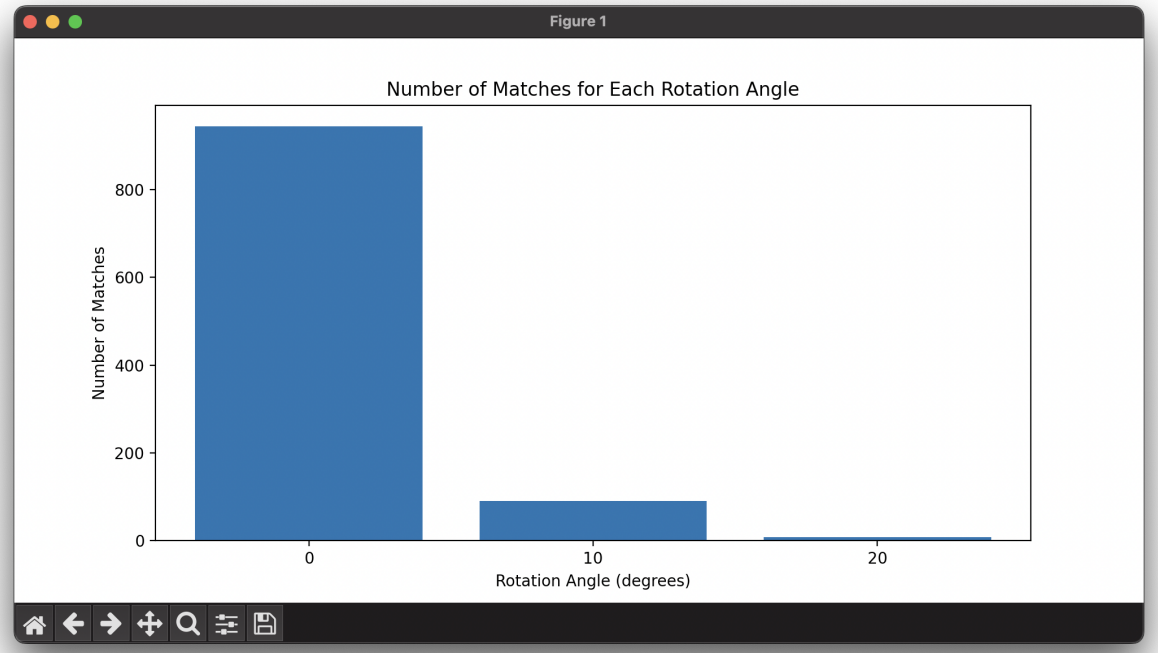

Figure 3: Matches at first 3 orientations

The BRIEF Descriptor shows close to a 1000 matches at no rotation, close to a 100 matches at 10 degrees and extremely few matches at 20 degrees rotation. When we observe for all rotation angles upto 360 degrees we see that most rotation angles show extremely few matches and these matches increase at certain angles like 120, 180, etc. This shows BRIEF descriptor is sensitive to rotation. This might be because BRIEF works by looking at specific pairs of pixels near key points in the image to see how bright they are. These pairs are set in one place and don't move even if the image is turned. So, when you rotate an image, these pixel pairs end up in different spots, which messes up the way they were originally compared.

Q3.9 Putting it together

At this point you should notice that although the image is being warped to the correct location, it is not filling up the same space as the book. Why do you think this is happening? How would you modify hp_cover.jpg to fix this issue?

We can see, in Figure 4 below, that the image is not filling up the whole cover, this might be because the original sizes of hp_cover and cv_cover are different and the matches are according to the homography of cv_cover in cv_desk.

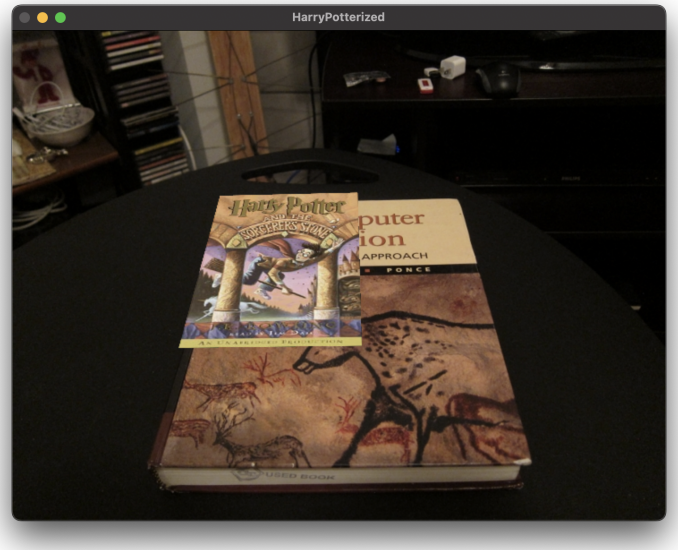

Figure 4: Composite image without scaling

My solution to this is scaling hp_cover according to the size of cv_cover

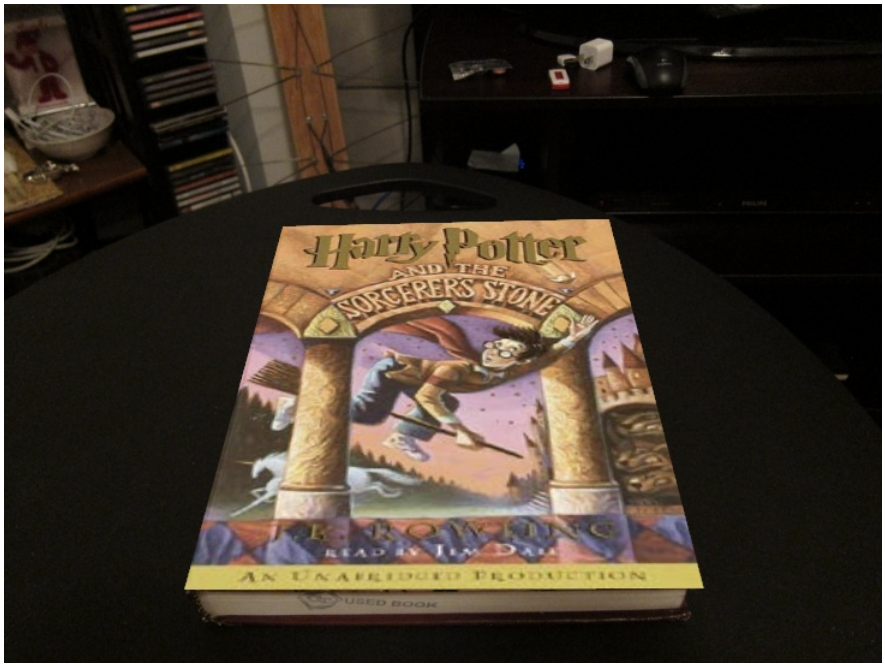

Figure 5: Final Composite image (iterations=200, threshold=2)

Submitted by Aryan Dawer on February 5, 2024.