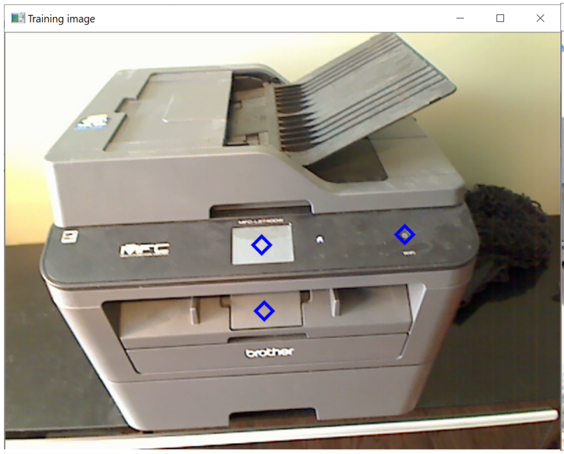
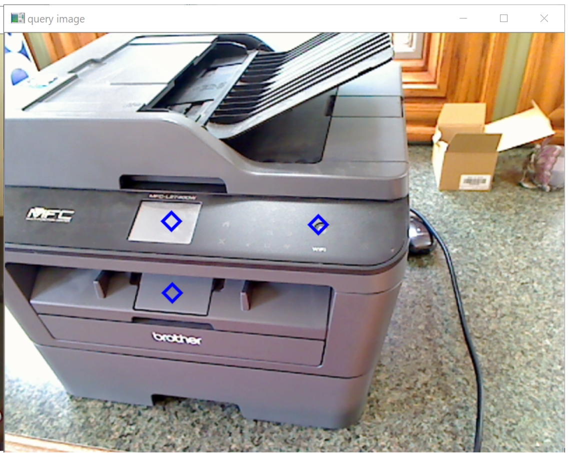
Lab 9: Object Annotation

Due Nov 2, 2020 by 11:59pm **Points** 10 **Submitting** a file upload

Overview

The goal of this exercise is to place one or more "annotation points" on a training image of an object, and automatically show those points in the correct place on the object in subsequent query images. For example, in the left figure below, I manually placed three annotation points on a training image of a printer, showing semantically meaningful locations. The right image shows those annotations correctly placed on a query image.





The approach we will take is to extract features (keypoints and descriptors) from the training image, then match them to features from the query image. We then fit a 2D affine transformation to the potential matches and use RANSAC to eliminate outliers. Once we have the affine matrix, we use it to transform the locations of the annotation points in the training image to their locations in the query image.

On the course website is a training image **printer_001.png** (https://elearning.mines.edu/courses/25410/files/2328358/download?download_frd=1) . Pick one or more points on this object to annotate and manually determine their (x,y) locations. To do this, you can use any program that displays the location of the mouse, such as "Paint" in Windows.

Write a program that extract features (keypoints and descriptors) from the training image. Then for each query image, do the following:

- Extract features from the query image.
- Match features between the training and query image. Use the "ratio test" to eliminate ambiguous matches.
- Fit a 2D affine transformation to the matches, using RANSAC to eliminate outliers.
- If the number of inlier matches exceeds a minimum threshold, then the object has been found. Apply the affine transform matrix to map the annotation points from the training image to the query image, and display the query image with the annotation points in the appropriate places.

Note – a critical parameter in this algorithm is the threshold number of inlier matches. If the threshold is too low, you may have "false positives", meaning that you detect the object where it is not actually present. If the threshold is too high, you may have "false negatives", meaning that you miss detecting the object even if where it is actually present.

Apply your program to each image in the folder **query_images.zip** (https://elearning.mines.edu/courses/25410/files/2328367/download?download_frd=1) . Adjust the threshold so that your program achieves the highest accuracy on this data, measured as the number of correctly classified images.

Upload:

- Your python program
- A pdf file containing the following:
  - The threshold that you used.
  - Which images were classified correctly.
  - Show the training image with your annotations.
  - Show at least two examples of query images, with the annotations correctly placed.