CS 423/523 Introduction to Computer Vision - Assignment 2

Due: December 14th, 2018 @ 23:55

Homography Estimation

Description:

In this assignment, your objective is to estimate the homography matrices between images of the same scenes. You can find the dataset for this assignment from (http://www.robots.ox.ac.uk/~vgg/research/affine/). **Make sure that you download all of the scenes at the bottom of the page and read the data description.** The dataset consists of six images of eight scenes with various changes such as, viewpoints, zoom and rotation, blur, lighting and jpeg compression. Your pipeline should be as follows, detect key points on two images of the same scene, create descriptors for detected key points, match the descriptors from two images and estimate the homography matrix from the matched key points.

Details:

You must choose two of the following keypoint detectors:

- Harris
- Shi-Tomasi
- FAST
- Star

You must also choose two of the following descriptors:

- SIFT
- SURF
- BRIEF
- ORB

To decide two descriptors from two images are the same, you need a distance metric L2 norm and a threshold λ. If the L2 norm of the two descriptors are smaller than λ, then these two descriptors are a match (belongs to same location). Formal definition is given as follows, where d_1 and d_2 belongs to descriptors from image 1 and 2.

$$match(d_1, d_2) = (\lambda > L2Norm(d_1, d_2))$$

After you have obtained a set of matched key points M_2^1 between image 1 and 2, you need to apply RANSAC **manually** to estimate the 3x3 homography matrix $\widehat{H}_2^1$ from image 1 to 2. **You need to apply this pipeline for each scene and every image** meaning that you need to obtain $\widehat{H}_2^1$ where n $=$ 2,3,4,5,6 for each scene.

The threshold value λ is the hyper parameter for your pipeline. Now you need an empirical method to choose a good λ value for your pipeline. You will find this via brute force search and utilizing precision-recall curve. Given a λ, you need to first apply estimated $\widehat{H}_2^1$ and the ground truth H_2^1 (E.g. H1to2p in dataset) to all key points in image 1 to map them on to image 2. Now you have the estimated $\widehat{X}_2^1$ and the ground truth mappings X_2^1 of the key points on image 2 coordinate system. With these two sets, you need to calculate False Positive, False Negative, True Positive values and draw the precision recall curves with respect to λ.

Iff an estimated point $\hat{x}_n \in \widehat{X}_2^1$ is closer than 2 pixels to $x_n \in X_2^1$ →True Positive
Iff an estimated point $\hat{x}_n \in \widehat{X}_2^1$ is closer than 2 pixels to $x_{m \neq n} \in X_2^1$ →False Positive
Iff an estimated point $\hat{x}_n \in \widehat{X}_2^1$ is not closer than 2 pixels to $x_n \in X_2^1$ →False Negative

You need to brute force the search for the λ that equally emphasize precision and recall (search for maximum recall and precision at the same time). An example of the graph you should be obtaining is given in figure 1[1].

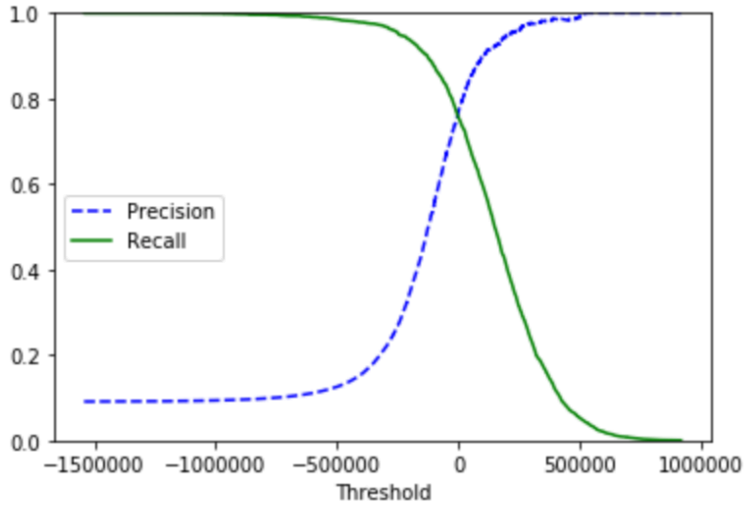

Figure 1 Precision and recall curves with respect to threshold.

You must write a detailed report on your results, pipeline, RANSAC algorithm and comments of homography estimation performances on different changes. The report must contain images of found and matched keypoints between two images (check OpenCV tutorials on SIFT).

Important notes:

- **Python 3.6 or higher is a must.**
- **Test your code before submitting, it must not crash.**
- **You may only use OpenCV, Numpy and Matplotlib.**
- **Your Assignment will be disregarded if your code does not work or a report is not submitted!**
- **Your Assignment will be disregarded if you use any other library than the ones stated above!**
- **READ THE DOCUMENT CAREFULLY.**
- **Your script should do the brute force search and draw the graphs given the enclosing folder of the dataset.**
- **Do not rename folders of the dataset, graf, graf-2 etc. is ok.**

Grading:

Your grade will be penalized 10 out of 100 points for each minor fix before running the code regarding the assignment details.
10%: Key point detection
10%: Descriptors
20%: RANSAC
20%: Precision Recall Curve
30%: Report

Submission Information:

Send all your source codes and report to the LMS. Your code should be clean and easy to read by possessing the following properties;

- *Clean structure:* The overall code should be neatly organized, where the related statements are grouped together with enough spacing among them.
- *Appropriate use of comments:* There should be comments explaining what the program, and different groups of statements are supposed to do. Don't overdo it.

[1] Figure taken from: https://www.kaggle.com/sikora507/mnist-binary-classifier-precision-and-recall

- *Meaningful and consistent variable naming:* The names of variables should be meaningful with respect to the purpose and usage of these variables.

Submission: By uploading your code and report to LMS as a single ZIP archive. No other methods (e.g., by e-mail) accepted. (You may resubmit as many times as you want until the deadline).
Warning: DO NOT SHARE YOUR CODE WITH OTHERS. Your programs are checked and compared against each other using automated tools. Any act of cheating will be punished severely.
Also:

- Name your archive file uploaded exactly as requested. Your archive file must be named as **<NAME>_<SURNAME>_<STUDENTID>.zip.**
- Make sure that your program runs and gives the **expected output.**
- The first lines of your code must include your name, surname, student number, and department as a **comment**. An example comment is as follows:
  / John Smith S0001 Department of Computer Science */*
- **Don't include your image and video files** in your archive

Good luck ☺