Assignment #2
10/01/2020

Prepared
for
Prof. Morato

Prepared
by
Abedin Sherifi

Introduction:

This homework assignment was divided into the following three sections:

> **a.** Implement a Harris' corner detector from scratch in python.
> **b.** Take four pictures of a scene from different angles and positions. The pictures have to have an overlap area between each other. Use your version of Harris' corner detector to stitch these four images and generate a single image.
> **c.** Crazy video: Use a handheld camera and record 10 seconds of video at 30 frames per second. While recording the video, change the attitude and position of the camera randomly (feel free to be creative...). Use your version of Harris' corner detector in each frame of the video and generate a new video at 10fps showing the corners detected and a counter with the number of corners detected in each frame.

Summary of Results:

Part A:

In the first part of the homework assignment, the Harris Corner Detector algorithm was applied to a picture. A brief overview of the steps involved in this algorithm is given below:

→ First the input image was read and was converted from RGB to Gray.
→ Gaussian filter was applied with a 9x9 kernel and sigmax/sigmay of 3.
→ X and Y gradients were computed using the sobel filter with kernel size of 9x9.
→ The components of the matrix M were calculated. These component are Ix^2, Iy^2, and IxIy.

$$M \qquad \underset{x,y}{} w(x,y) \begin{matrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{matrix}$$

→ The corner response function is calculated.

$$R = det(M) - \alpha \cdot trace(M)^2$$

Reminder:

$$det\left(\begin{bmatrix} a & b \\ c & d \end{bmatrix}\right) = ad - bc \qquad trace\left(\begin{bmatrix} a & b \\ c & d \end{bmatrix}\right) = a + d$$

→ The last step of the algorithm is the execution of non-maximum suppression. This is the step were we had to empirically choose a threshold value above which a corner is detected.

OpenCV libraries were used throughout the code. Lastly, a corner count is displayed on the image.

The output file from running the Harris Code Detector is shown below:

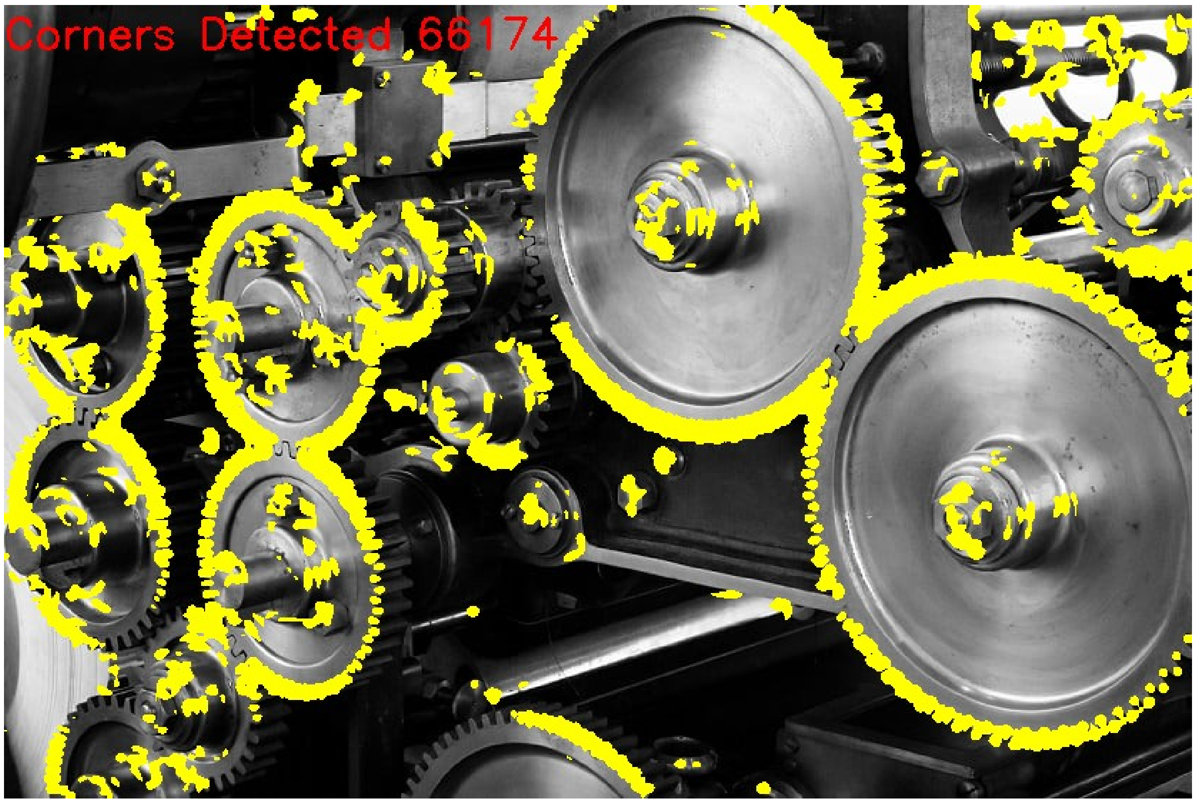

Figure 1. Harris Corner Detector on an image.

Part B:

In this part, we were supposed to use the Harris Code Detector to generate the keypoints and then match the descriptors of these keypoints. Due to work and personal reasons, I feel like I did not have enough time to spend on this step. If I had more time, I would have wanted to use the Harris Corner Detector code and stitch multiple images instead of just 2 images. The work I performed for image stitching did not involve the Harris Corner Detector but it involved the sift feature extraction. I also only stitched two images instead of taking the code one step forward and to be able to stitch multiple images.

The steps involved in stitching were the following:

- → First step is the input of an image and converting that image from RGB to GRAY.
- → The second step computes the descriptors and keypoints on both images using SIFT.

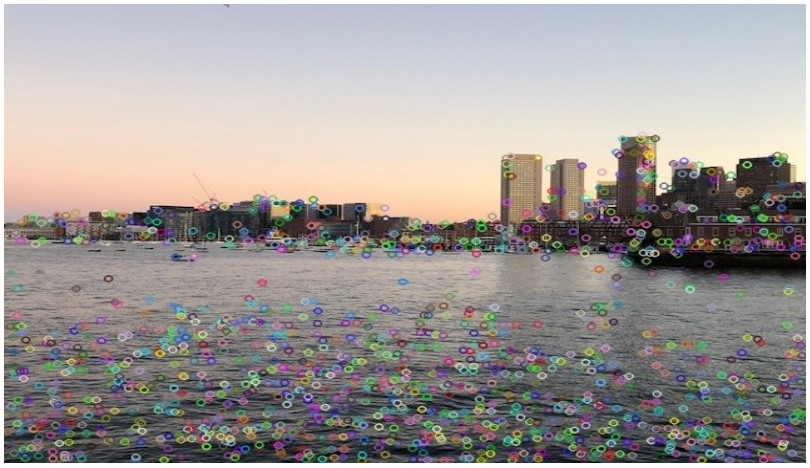

Figure 2. The SIFT keypoints on an image.

→ The third step uses the FLANN method for matching the most similar descriptors from both images.
→ The next step applied is filtering of only the best matches with shorted distance ratio.
→ Next, RANSAC is run and estimated homography is computed and image alignment performed.
→ The last step of this code is image wrapping and the stitching of the images.

An example of stitching of two images is shown below:

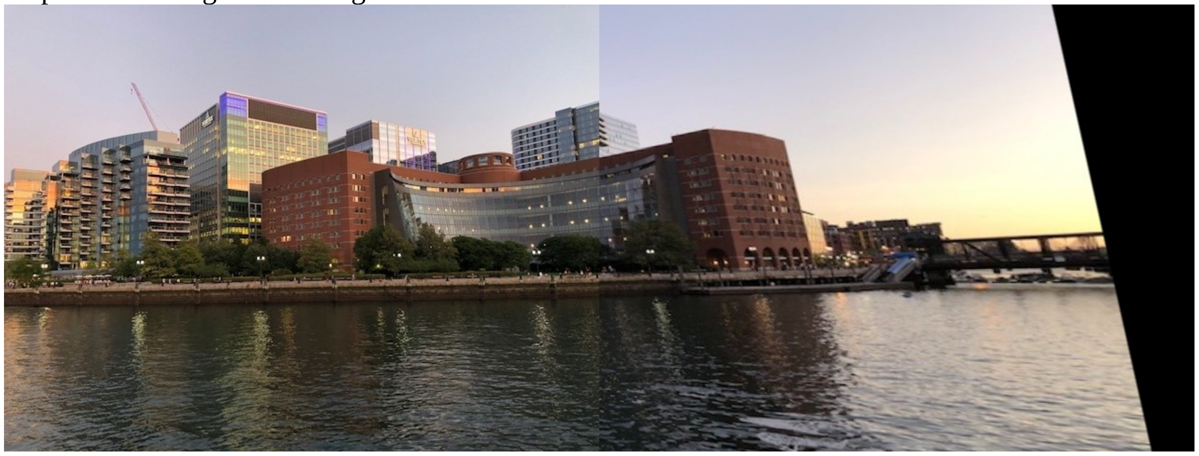

Figure 3. Image stitching on two images.

Part C:

In this third part, we were supposed to take a video at 30 fps and then detect the Harris Corners with the corner count per frame and display the video at 10 fps. Instead of taking a video with my phone, I instead took a video with my webcam and applied the Harris Corner Detector per frame.

The same steps are applicable to this code as they were for the Harris Corner Detector. The only difference here is that we capture webcam frame by frame. We apply the Harris Corner Detector on each frame and we also display the corner count and the corners themselves for each frame on the video. We record a video using webcam at 30 fps and then we output a video with the Harris Corners and the count at 10 fps.

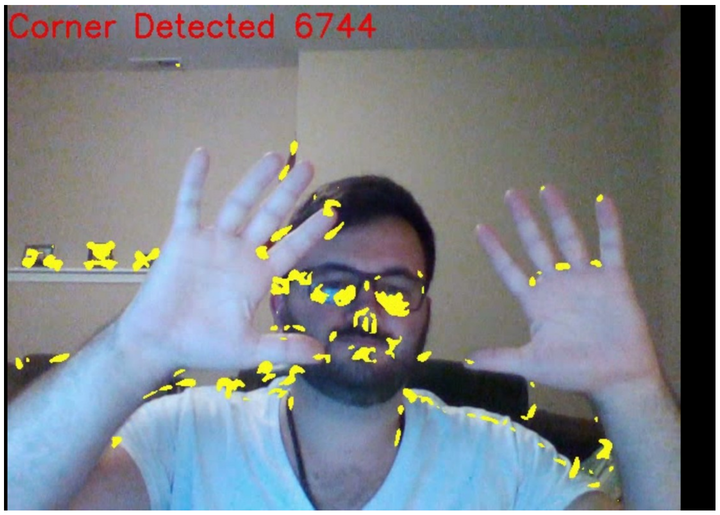

Figure 4. Harris Corners on a video frame.