Lab 6 homework report

Course: Computer Vision 2020
Francesco Pham (1234004)
Miroljub Mihailovic (1163471)

In this report we are covering our lab experience on object recognition and tracking. During program execution you will be prompted to type in the path to the folder containing the objects images and the video.

The project structure

The objects and the video are loaded inside the main function inside *main.cpp* which in turn calls `mainHomework6` inside *homeword_6.cpp*.

In `mainHomework6` we firstly locate each object in the first frame of the video by calling `detectAndMatchKeypoints` and we assign a random color for each object. To detect the keypoints we tried both ORB and SIFT feature detector, since the latter gave much better results we went with SIFT.

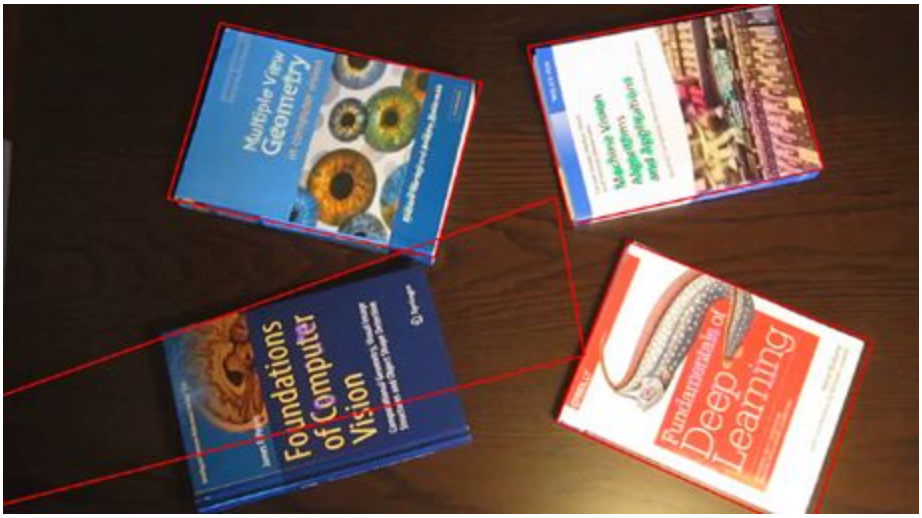
Then inside a while loop we capture a new frame and recompute the new keypoints positions based on the optical flow inside `trackObjects` function. Finally, we call `drawKeypointsAndRectangles` to draw the keypoints and the bounding rectangles around the objects.

At the top of *homeword_6.cpp* there are two constants DISTANCE_RATIO_THRESHOLD and RANSAC_REPROJ_THRESHOLD used to regulate the discarding of the outliers based on the distance between matches and the Ransac threshold, after trying different settings we finally kept 3.0 for both.

Our results

In the final result we can see that the keypoints are correctly detected on the objects and the features are accurately tracked across the frames.

The results that we obtained using ORB feature detector are unsatisfying because we had to increase the max features in order to match the four books with the first frame. This procedure increases the time of execution of the program however, the best result obtainable using this algorithm was still imperfect, as could be seen in the figure below.



At first we tried to track also the corner points of the rectangles but that resulted in the points to be attached to the wrong book or to the hand that covered the corner as we

can see in the figure below. So using `findHomography` to estimate the object translation and rotation based on all the keypoints was the best way to go.