37212-cwk2-S-Object_detection_k83954ao

Andrei N. Onea

May 17, 2021

Experiment

This assignment attempts to match an object from a reference image, in several benchmark images. The subject is the person Bernie Sanders, and the benchmark images provided represent different variations in rotation, scale, illumination, blurring etc. to test the performance of the object detector. The implementation has three parts. Part 1 deals with feature detection using own implementation of the Harris corner detector, Part 2 deals with feature description, and descriptors are constructed using the OpenCV built-in ORB framework, and finally Part 3 deals with feature matching, using own implementation of sum of squared differences with ratio test for discarding ambiguous matches. We will compare some results and provide our thoughts on the experiment.

Part 1: Feature detection

The main steps for feature detection are: blur the gray-scaled image, compute the image derivatives using Sobel operators, compute the combinations of the image derivatives, apply a Gaussian blur on the combinations, compute the matrix, compute the cornerness function $R = det(M) - \alpha \times trace(M)^2$ for each pixel and finally filter the corners using own implementation for non-maxima suppression with thresholding. Keypoints are then created from the filtered values. In Figure 1 we see the evolution of the number of keypoints as the threshold value changes. We notice an exponential decrease, as there are many fluctuations in the values at smaller scale, but as the values increase, they are increasingly more the same, so the constant lines at the end appear.

Our choice of threshold is 80. We chose the value empirically, as we tried to minimize the number of features detected outside of Bernie's shape. The Appendix contains all the keypoints detected in all images. Discussion on the performance of the feature detection implementation will be made in Part 3.

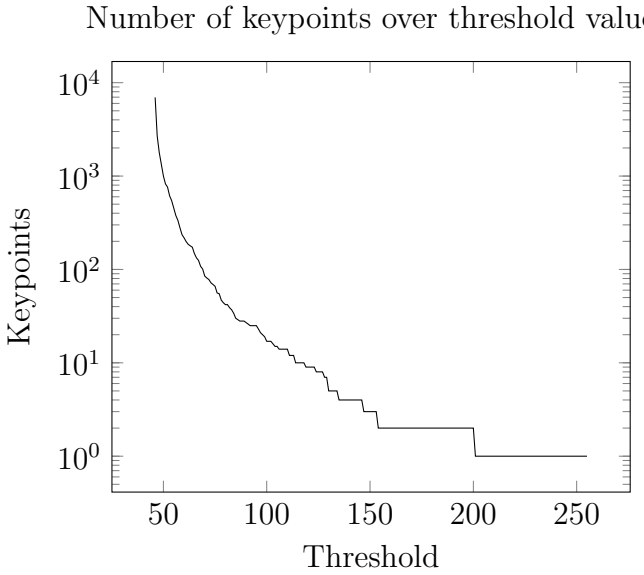

Figure 1: Variation in the number of keypoints as the threshold value increases

Part 2: Feature description

For this step, we used the built-in ORB framework in the OpenCV package. Figure 2 has side-by-side the keypoints detected by our implementation with the keypoints detected by ORB. We

can see that in this regard, the performance of your feature detector is almost indistinguishable from the ORB detector.

Figure 2: (left) our implementation, (right) ORB detect function

Part 3: Feature matching

The last step is to try matching the features from the reference image with features from the benchmark images. This is done by computing the sum of squared differences between two feature windows. This is done for every combination of features and then the smallest distance is picked. Because this might return ambiguous results, the second smallest distance is also picked then a ratio between the first and the second is made. If this ratio is close to 1, then the match is discarded. We chose 0.9 as the threshold value for the ratio test, and 200 as the threshold value for the maximum distance for which a match is selected.

Rotated image

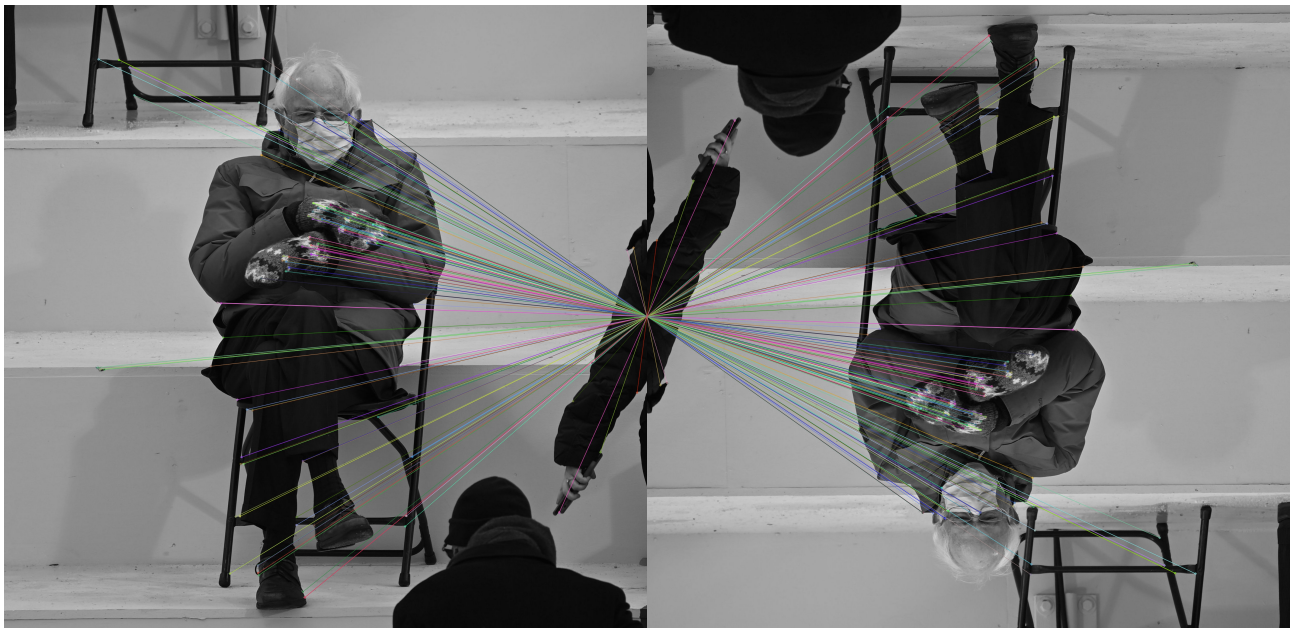

Figure 3: Rotated image

Because the only variation is in rotation, the matcher gets all the features right.

Scaled image

Figure 4: Scaled image

As it will be the case with all scaled benchmark images, the matcher does not find any result. That is because our basic implementation is not scale invariant, meaning that it cannot detect changes in scale.

Blurred image

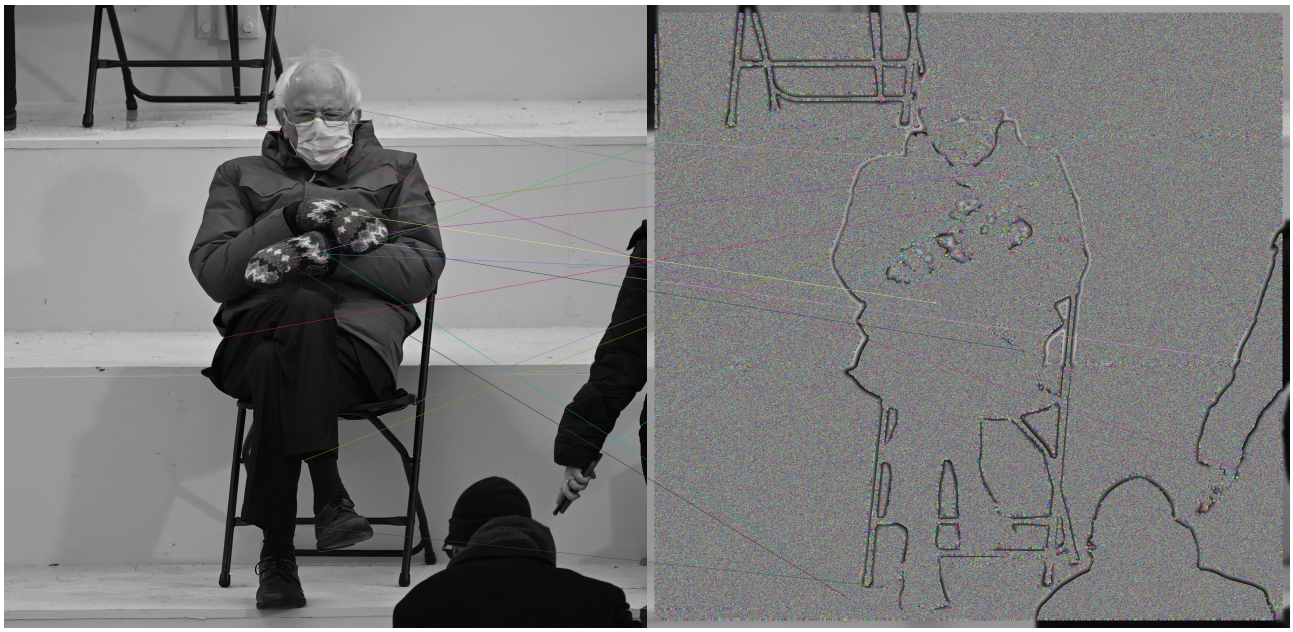

Figure 5: Blurred image

Because the image is already quite blurred, fine detail is lost and the detector takes any flat surface as a corner, and only distinguishes flat surfaces from edges. That is why there are so many keypoints. However, there is one single feature that is correctly matched, that is one point on the glove, here represented in yellow.

Noisy image

Figure 6: Noisy image

Because the image is noisy, pixels are misinterpreted. There is a match, again on the glove, represented in cyan.

Pixelated image

Figure 7: Pixelated image

This example is similar to the noisy image above, but the pixelated pattern is less impactful, thus more matches are correctly found.

Similar characters and scaled image

Figure 8: Similar characters and scaled image

Surprisingly, there are two matches on the mask and the glove which correspond perfectly. The others are misinterpreted, as expected with images that have variations in scale.

Many characters and scaled image

Figure 9: Many characters and scaled image

A better example of why images with variations in scale do not produce good results. The distances are usually off in these cases and get discarded by the threshold test. Although features are present, no match is found.

Brighter image / Darker image

Figure 10: Brighter image

Figure 11: Darker image

In both cases, the changes in intensities confuse the matcher greatly.

Conclusions

Our implementation performs poorly when faced with variations. The sigma for Gaussian blurring affects how sharp the features are placed. The SSD threshold can create a lot of wrong matches even with the ratio threshold set high, especially in the case where the benchmark is at a different scale. It has been empirically found that most images do not produce low ratios between the first and second match, even for correct matches. Another way to choose the threshold for picking the keypoints algorithmically would have been to pick the top 30% and select the lowest value as the threshold value, quantitatively choosing the threshold. Instead, we opted to use an empirical method. We believe finding good parameters to make the model perform well is a very time-consuming task for the level of experience that we possess in this moment, so we conclude by being satisfied that we have reached this far. This is an excellent point from which to start further studies in Computer Vision.

Appendix

Figure 12: Reference image

Figure 13: Rotated image

Figure 14: Scaled image

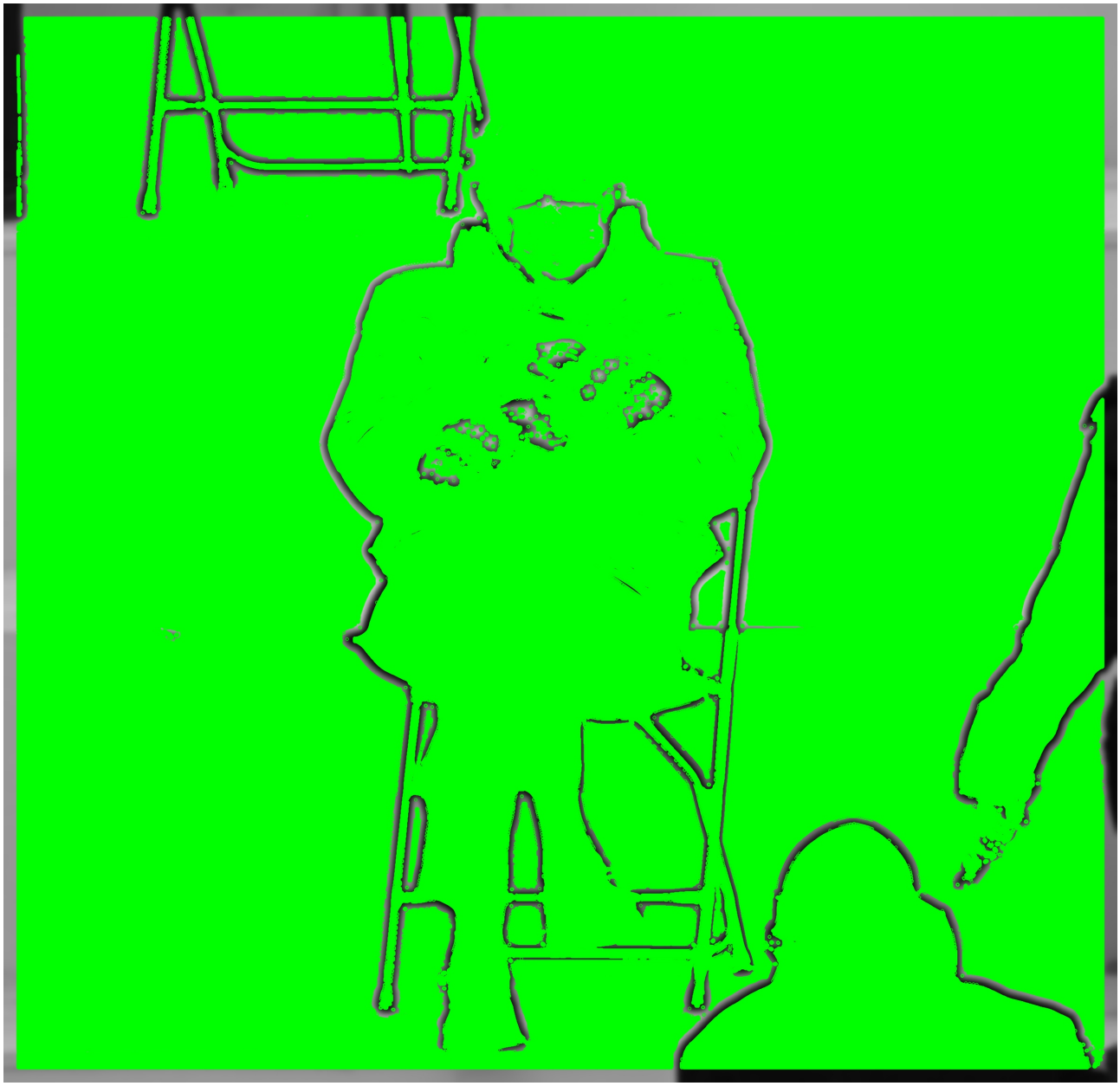

Figure 15: Blurred image image

Figure 16: Noisy image

Figure 17: Pixelated image

Figure 18: Similar characters and scaled image

Figure 19: Many characters and scaled image

Figure 20: Brighter image

Figure 21: Darker image