Basic Computer Vision- Final Project
Report of Marker-based augmented reality

Xuan ZHOU
Institut Polytechnique de Parise
xuan.zhou@telecom-sudparis.eu

Part I. Theoretical study

I.1 SIFT (Scale Invariant Feature Transform)

SIFT algorithm is a local feature description algorithm in the field of image processing. The method was proposed by Canadian professor David G. Lowe in 1999. SIFT algorithm is not only has only scale-invariant, but also can still get better detection effects when rotating the image, changing the image brightness, and moving the shooting position. SIFT has better stability and no deformation, can properly rotate, scale scaling, and change of brightness, and can be somewhat independent of perspective change, affine transformation, and interference of noise. It has good distinguishability and can quickly and accurately distinguish information for matching in the massive feature database and can generate a large number of feature vectors even if there is only a single object, which can quickly perform feature vector matching.

II. 2 SURF (Speeded Up Robust Features)

SURF algorithm is an improvement of SIFT, SURF improves the extraction and description of features in a more efficient way. The improvement of SURF is to use the Hessian matrix to transform the image, the detection of extreme values only needs to calculate the Hessian matrix determinant, and as a further optimization, a simple equation can be used to find the Hessian determinant approximation, using box blur filtering (box blur) to find the Gaussian blur approximation.

III.3 ORB (Oriented FAST and Rotated BRIEF)

ORB algorithm is a fast feature point extraction and description algorithm which was proposed by Ethan Rublee, Vincent Rabaud, Kurt Konolige and Gary R. Bradski.ORB algorithm is divided into two parts which are feature point extraction and feature point description. Feature extraction is developed from the FAST (Features from Accelerated Segment Test) algorithm, and feature point description is improved from the BRIEF (Binary Robust Independent Elementary Features) feature description algorithm. ORB features are a combination of the FAST The most important feature of the ORB algorithm is its fast computational speed. This is firstly due to the use of FAST to detect feature points, which is notoriously fast as its name implies. Again, the descriptors are computed using the BRIEF algorithm, and the unique binary string representation of the descriptors not only saves storage space but also greatly reduces the matching time.

Part II. Descriptor matching

II.1. BFM (Brute-Force matcher)

Brute-Force matcher is simple. It takes the descriptor of one feature in the first set and is matched with all other features in the second set using some distance calculation. And the closest one is returned.

For BF matcher, first we have to create the BFMatcher object using cv.BFMatcher(). It takes two optional parameters. The first one is normType. It specifies the distance measurement to be used. By default, it is cv.NORM_L2. It is good for SIFT, SURF, etc (cv.NORM_L1 is also there). For binary string-based descriptors like ORB, BRIEF, BRISK, etc, cv.NORM_HAMMING should be used, which used Hamming distance as measurement. If ORB is using WTA_K == 3 or 4, cv.NORM_HAMMING2 should be used.

Second param is a boolean variable, crossCheck which is false by default. If it is true, Matcher returns only those matches with value (i,j) such that the i-th descriptor in set A has the j-th descriptor in set B as the best match and vice-versa. That is, the two features in both sets should match each other. It provides a consistent result, and is a good alternative to the ratio test proposed by D.Lowe in SIFT paper.

II. 2 Matching methods & filtering methods

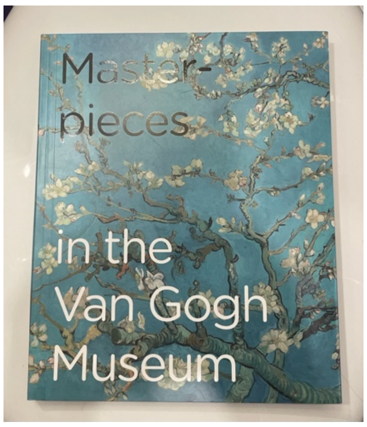
Match() and knnMatch() are keypoint matches for each matcher. They have a different division of labor, match() finds the best match, while knnMatch() will return the best k match for each descriptor.

II. 3 Alignment & Augmentation

In OpenCV, the routine findHomography() will compute and return the perspective transformation between them H. We can specify the RANSAC robust algorithm in this method to try different random subsets to estimate the single-strain matrix. warpPerspective () applies the perspective transformation to the image. By placing the target image with the target image, an image transformation can be performed.

Part III. Experiment

In this experiment, I compared the detection and matching operation times of the SIFT, SURF and ORB algorithms, each detector pointer was created using default parameters that are supposed to be optimal. I compared simple objects in light and dark cases separately. Then the complex objects were compared under light and dark cases. The first is in the dark case, as shown in Figure 1, and the results of the three algorithms are shown in Figure 2.



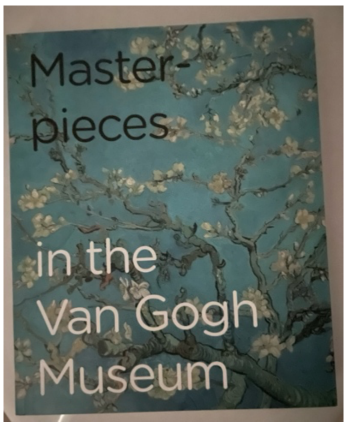
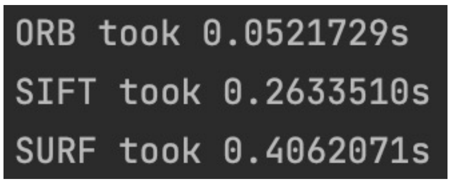
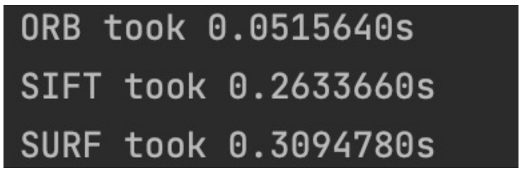


Figure1. Simple objects in light and dark cases respectively



```
ORB took 0.0521729s
SIFT took 0.2633510s
SURF took 0.4062071s
```



```
ORB took 0.0515640s
SIFT took 0.2633660s
SURF took 0.3094780s
```

Figure 2. Running time of simple objects in light and dark respectively.

Figure 3 shows the comparison of complex objects in light and dark cases, and Figure 4 shows the results of the three algorithms for the comparison in Figure 3.

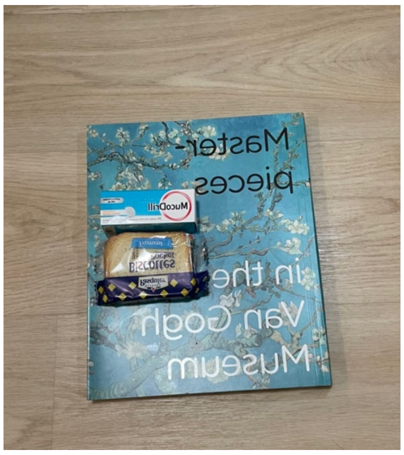
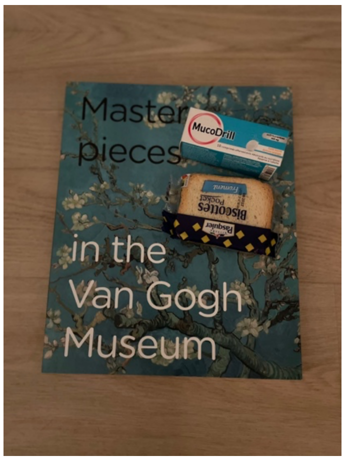
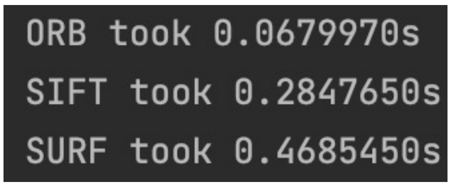
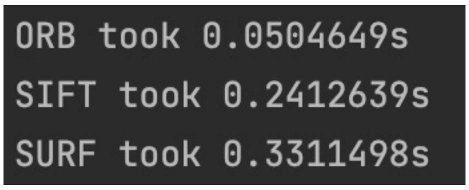



Figure 3. Complex objects in light and dark



```
ORB took 0.0679970s
SIFT took 0.2847650s
SURF took 0.4685450s
```



```
ORB took 0.0504649s
SIFT took 0.2412639s
SURF took 0.3311498s
```

Figure 4. Complex objects in light and dark

By comparing the running time of simple and complex objects under light and dark respectively, we can find that the ORB algorithm is the fastest, so I choose the ORB algorithm in the experiment. The authors used the BFM algorithm and KNN algorithm for matching and screening because they found that the frame rate of the video was higher and the enhanced image display was stable during the experiment, so they did not experiment with other algorithms, so they used these two algorithms as the final solution for the module.

Part IV. Final solution

In this experiment, I use ORB for image descriptor crawling, BFM for descriptor matching, and KNN algorithm for description matching filtering.
First, I get the shortest distance by traversing the well-matched descriptors, and then I filter out the bad matches by comparing the distance of each pair of descriptors with that minimum distance to get the good matching pairs. Finally, I set ransacReprojThreshold to 4 when I calculate the computational pivot matrix. this is my final solution.

Part V. Conclusion

In this experiment, I compared the results of different objects in different situations, which made me a deeper understanding of marker-based augmented reality, I learned how to compare different algorithms and choose the best one, and I also have a clearer understanding of what my professor explained in class.