Alp G. Atakav & Arman Fidanoglu, aga3718 & af5918, 01523868 & 01512561

1. Data Collection

This report presents a collection of tasks performed on images, including keypoint detection, homography and fundamental matrix estimation, stereo rectification and depth estimation. To accomplish these tasks, two forms of data are required: a 'HG' dataset which consists of 50 affine transformations (small changes in zoom and rotation) of one image, and an 'FD' dataset which presents 50 different pictures of the same scene taken from different perspectives.

For both datasets, the backdrop was chosen to be a 3D grid covered in checkerboard patterns. Then, an appropriate textured object (small toy) was placed in the middle of the grid. The toy's shape, size and textures aid the effective detection and matching of keypoints. Both datasets contain images with and without the object in the grid.

Both image sequences were constantly updated during the experiments, as new requirements were identified for each task. 10 samples from the HG and FD datasets are located in appendix 6.1 and 6.2, respectively. All images used were taken with a Google Pixel 4 XL smartphone. To avoid automatic corrections to the images by the phone's computational photography system, all images were taken in RAW mode.

2. Keypoint Correspondences

This task aims to compare the quality and quantity of keypoint correspondences found manually and automatically. In both cases, an image pair was chosen from the FD dataset. Then, keypoints were identified and matched. After this task was completed, all experiments were repeated on an image pair from the HG dataset, to obtain a quantitative measure for the performance differences.

To evaluate the performance of each method, two metrics were utilised:

1. Precision (P): The ratio of correct matches (True positives) to all matches (True positives + False positives)

2. Recall (R): The ratio of correct matches (True positives) to all keypoints that should have been matched (True positives + False negatives)

Method 1: Manual A Python program was written, which converts clicked points on the image pair to coordinates, which are then plugged into a keypoint matcher in OpenCV. Since manually selected keypoints do not have descriptors associated with them, a descriptor-based matcher like FLANN could not be used. Instead, a brute-force K-Nearest-Neighbour (KNN) matching algorithm was used with l^2 distance. Afterwards, Lowe's ratio test was applied to the array of matches to filter out implausible matches [2].

For this task, images from the HG and FD datasets were each tested with Lowe thresholds of 0.5 and 0.7. Figure 1 contains an example FD pair with 55 keypoints manually selected per image, and a threshold of 0.7 used for Lowe's ratio test.

Figure 1. Manual keypoint capture and matching on images from the FD dataset

Method 2: Automatic SIFT descriptors allow detection and description of local features in images [1]. Due to their robust and distinctive nature, they lead to exceptional matching performance [3]. Thus, this descriptor type was selected for automatic keypoint matching. To match the descriptors, a FLANN KNN matcher with l^2 distance was used, following OpenCV documentation, which states that this matcher/distance combination works best with SIFT descriptors.

Again, Lowe thresholds of 0.5 and 0.7 were used on images in the FD and HG datasets. However, this time each experiment was repeated with 150 and 1000 keypoints per image, which put the precision into perspective. Figure 2 contains an example pair from FD with 150 SIFT keypoints per image and a threshold of 0.7 for Lowe's ratio test.

Performance Comparison Table 1 demonstrates the performance differences between the manual and automatic (SIFT) methods, as well as the effect of the number of keypoints per image (kp) and Lowe threshold (th).

Figure 2. Automatic keypoint capture and matching on images from the FD dataset

Method, kp, th	FD	HG
SIFT, 150, 0.5	20.5%, 6%	67.1%, 35%
SIFT, 150, 0.7	56.4%, 20%	77.9%, 47%
SIFT, 1000, 0.5	N/A, 3%	N/A, 17%
SIFT, 1000, 0.7	N/A, 12%	N/A, 30%
Manual, 55, 0.5	32.7%, 31%	0%, 0%
Manual, 55, 0.7	69.9%, 54%	4.4%, 4%

Table 1. Average recall and precision (R, P) values of each method on image pairs from each dataset

To compute the R and P values in table 1, the true positives (correct matches) and false negatives (keypoints that should have been matched) were found by both code and inspection. Furthermore, precision calculations were based on the number of matches before applying Lowe's ratio test. This is the reason why images with 1000 keypoints do not have recall values - since it was not possible to perform inspection. This table demonstrates a few points.

- The automatic method works about 2.3 times better on the HG dataset which is expected, as the lack of perspective changes results in identical descriptors.
- Manual keypoints result in much better performance, due to intentionally being placed in the same, very distinct areas on both images.
- Increasing Lowe's threshold by 40% results, on average, in a 50%+ precision increase, as a higher threshold means matching less similar keypoints. This leads to an increase in the number of true positives, and thus the recall.

Finally, manual matching on the HG dataset failed, likely due to sub-par keypoint selection. In consequent attempts, this is expected to be greater than its FD counterpart.

3. Camera Calibration

To calibrate the smartphone camera, OpenCV's inbuilt functions were used. These functions require a flat checkerboard pattern, and apply the pinhole camera model to estimate the camera parameters. To satisfy OpenCV's calibration requirements, a separate image dataset was constructed. The dataset used for this calibration task consists of 15 images of a single, 6x7 checkerboard pattern taken from various camera angles and positions. 9 samples from this dataset can be found in appendix 6.3.

The pinhole camera model describes extrinsic and intrinsic parameters, the former representing the transformation between the external world and the camera, and the latter representing internal parameters of the camera. These include focal length, optical centre and lens distortion.

To perform the calibration, the 3D and 2D locations of internal corners of each checkerboard image were extracted. Afterwards, a non-linear optimization algorithm was used to minimize the re-projection error between the 2D and 3D corner locations. Finally, the intrinsic camera matrix K was computed, along with an extrinsic matrix, which depends on the image. When multiplied, they make up P, which is the calibration matrix.

$$P = \begin{bmatrix} R \\ t \end{bmatrix} \times K, \text{ where } K = \begin{bmatrix} 3289 & 0 & 1508 \\ 0 & 3287 & 2044 \\ 0 & 0 & 1 \end{bmatrix} \quad (1)$$

Figure 3 shows the differences between the raw and corrected images. This difference is small due to computational improvements in smartphones, but still noticeable.

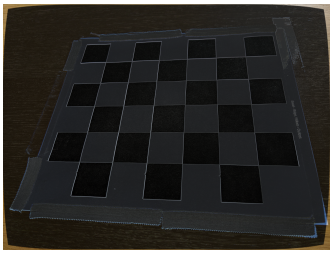


Figure 3. Difference between raw and corrected images: White areas (lines) are different, dark areas are similar.

4. Transformation Estimation

Task 1: Homography Matrix The homography matrix, **H**, relates two images through the transformation $x_2 = Hx_1$.

This task estimates the homography matrix between two images in the HG dataset. Because this matrix has 8 unknown parameters (degrees of freedom) and two axes per point, at least 4 points are required to estimate it - however, more points lead to better results. To compare the impact of points on this transformation, the same experiment was conducted with 4 and 12 correspondence points. The results of this experiment are shown in figure 4.

Figure 4 demonstrates that increasing the number of points greatly improves the quality and robustness of the projection.

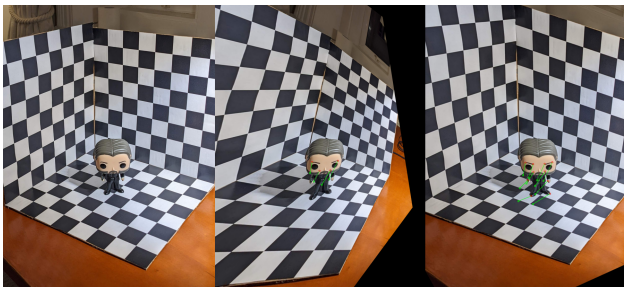


Figure 4. Destination Image, source image projected with 4 points and 12 points. Includes original and projected points.

Task 2: Fundamental Matrix Unlike the homography matrix, **F** relates points through the equation: $0 = x_2 F x_1$.

This equation represents the fact that the point x_2 lies on the line Fx_1, hence bringing their dot product to 0. Due to this property, points in one image in a stereo pair are associated with lines on the other. In order to estimate the fundamental matrix, two images from the FD dataset were utilised. After performing automatic keypoint matching and Lowe's ratio test, the epipolar lines were calculated as shown in figure 5. Please note that the images in this figure are flipped (L and R), since this position proved to be more space efficient.

In this image, the epipoles lie between the images. Although the authors of this report would have liked to include the epipoles in the image, the drastic scenery changes required to place the epipoles in the image led to wrong epipolar lines. Therefore, smaller changes were used, which placed the epipoles between the images.

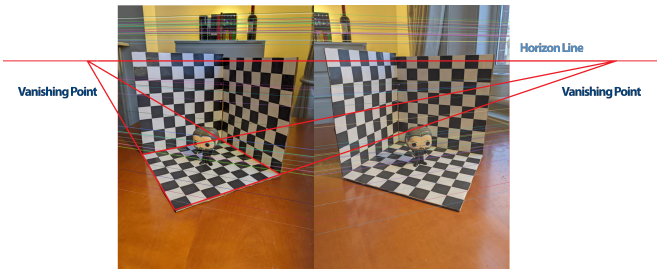


Figure 5. Points and corresponding epipolar lines on two FD images

Task 3: Outlier Toleration

The epipolar lines in figure 5 were calculated from keypoints filtered with a Lowe's threshold of 0.6. When the filter was removed, the epipolar lines were still correct, despite the 35 outliers - 32.7% of keypoints. This threshold was increased iteratively until the epipolar lines started converging at wrong spots - this happened to be a threshold of 0.75, which resulted in 63 outliers, at 13.3% of points. This is an interesting result which indicates that the ratio is not of essence, but the number of outliers should not exceed 60. This result may be explored further in later work.

5. 3D Geometry

Task 1: Stereo Rectified Pair For depth estimation, the stereo pair need to go through a transformation, allowing both images to lie on the same image plane. This aligns their epipolar lines, and makes it very easy to compute disparities, and therefore depth. Since the perspective of the images in figure 5 are very different, they are unsuitable for depth estimation. Thus, figure 6 shows a different FD image pair that was stereo rectified.

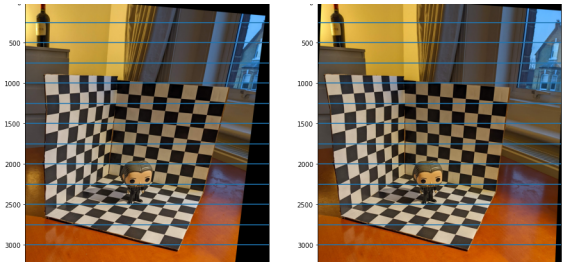


Figure 6. Rectified FD image pair

Task 2: Depth Map After the images in figure 6 are rectified, they can be used to compute a disparity map. However the camera focal length (in mm) and distance between camera positions between the two images is required for this task.

For the author's smartphone with $1.4\mu m$ pixels, the focal length was calculated from the intrinsic camera matrix from section 3, to be about 23.5 mm. Using this focal length and a guess for the distance between the cameras of $50mm$, a disparity map was constructed. This map is demonstrated in figure 7.

Figure 7. Disparity map between two FD images

Contrary to expectations, this disparity map does not represent the depth. The results did not change after several attempts with 30 different pairs of images, various focal lengths (including the manufacturer quoted value for the focal length, 27 mm), and a broad range of camera separations.

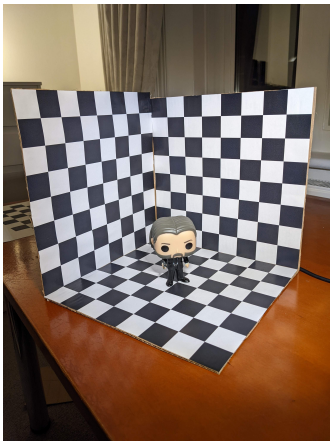
After succeeding to obtain a good disparity map on two OpenCV example images with the given parameters, it was concluded that the images used were not suitable.

References

[1] D. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157 vol.2, 1999.

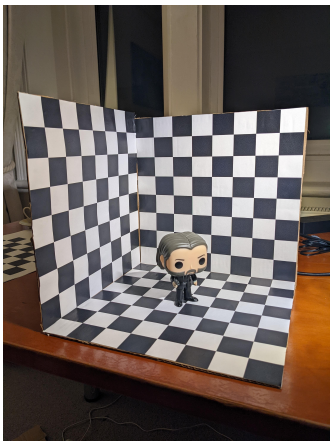
[2] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision 60*, page 91–110, 2004.

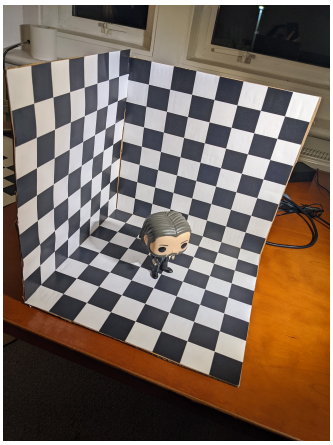
[3] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, 2005.

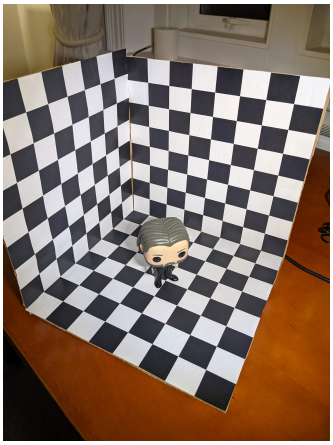


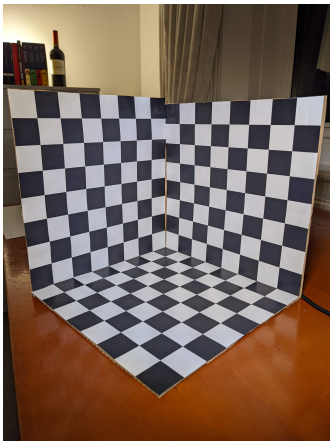
6. Appendix

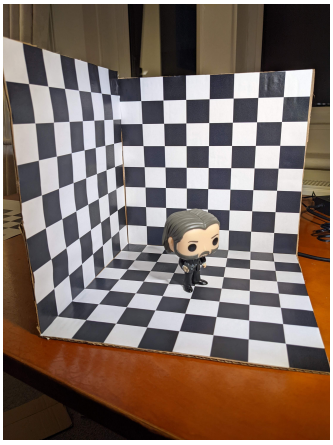
6.1. FD Image Dataset



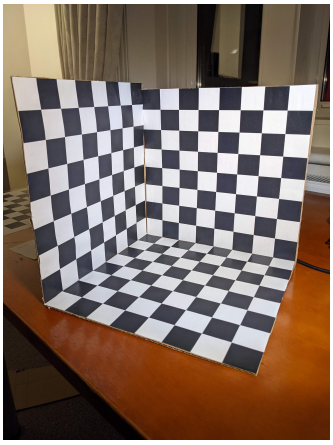


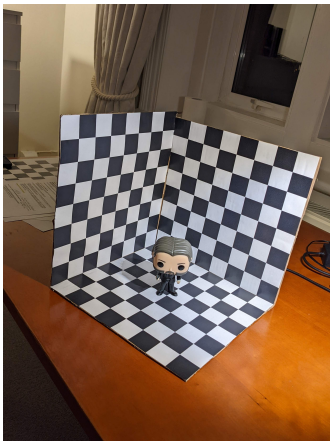


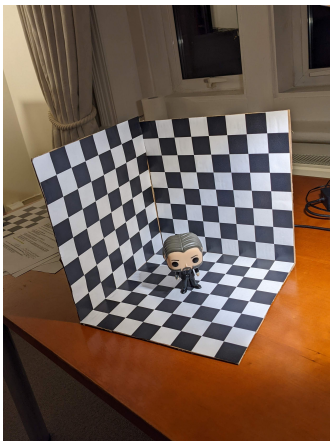


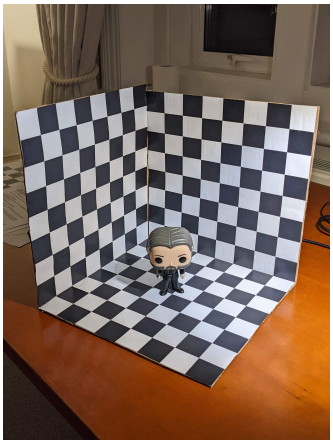


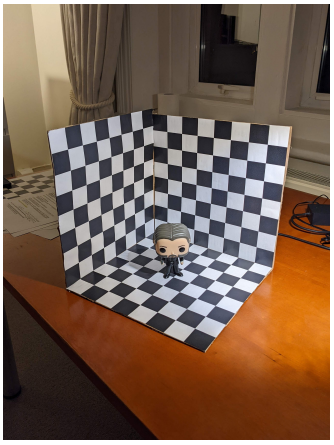
6.2. HG Image Dataset











6.3. Image Dataset for Camera Calibration