ENPM662

Project Four

SURIYA SURESH

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

Chapter 1

PROJECT 4

1.1 Pipeline

The pipeline followed for this project is:

- Loading Images from File
- Feature Extraction and Matching
- Normalizing Points
- Finding F Matrix Using RANSAC
- Finding E Matrix
- Decomposing E Matrix into Rotation and Translation
- Drawing Epilines
- Rectifying Image
- Correspondence
- Compute Depth Image

1.2 Feature Matching

For matching features, we load images from the data set.We initialize SIFT for feature detection and use BF matcher to match the features across the images.Then the matched points are sorted based on distance and the top 100 values are considered. Then we normalize the points[1] so that the points are on a uniform scale.The ceteroid of the points are found and then scale term is found to get the transformation matrix for norm.

$$\mathbf{s} = \frac{\sqrt{2}}{\left(\frac{1}{n}\sum_{i=1}^{n}\left(\tilde{u}_i^2 + \tilde{v}_i^2\right)\right)^{1/2}}$$

$$\mathbf{s}' = \frac{\sqrt{2}}{\left(\frac{1}{n}\sum_{i=1}^{n}\left(\tilde{u}'^2_i + \tilde{v}'^2_i\right)\right)^{1/2}}$$

Figure 1.1: Finding Scale Factor

$$T_a = \begin{pmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & -\bar{\mathbf{u}} \\ 0 & 1 & -\bar{\mathbf{v}} \\ 0 & 0 & 1 \end{pmatrix}$$

$$T_b = \begin{pmatrix} s' & 0 & 0 \\ 0 & s' & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & -\bar{\mathbf{u}}' \\ 0 & 1 & -\bar{\mathbf{v}}' \\ 0 & 0 & 1 \end{pmatrix}$$

Figure 1.2: The transformation matrix

1.3 Finding F Matrix

The fundamental matrix gives the relationship between two images taken using the same camera or different cameras at different positions which may have similar points projected in different manners.The matrix gives the relation between the points across the images.

We stack the points obtained in a A matrix and do SVD to obtain the F matrix from the last column of V transpose. We normalize F and then apply the rank condition for S being 2 and recalculate F after doing SVD for the initial F matrix to get the U,S and V matrices. To find F matrix, the normalized 8 point algorithm of Hartly was used to find the F matrix.

RANSAC is used to get the min 8 points needed to form the A matrix and is used to get the best F matrix over the entire data set and neglect noise from the F matrix.

$$\mathbf{A\,f} = \begin{bmatrix} x'_1 x_1 & x'_1 y_1 & x'_1 & y'_1 x_1 & y'_1 y_1 & y'_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x'_n x_n & x'_n y_n & x'_n & y'_n x_n & y'_n y_n & y'_n & x_n & y_n & 1 \end{bmatrix} \mathbf{f} = 0$$

Figure 1.3: The A matrix

1.4 Finding the E Matrix

The Essential matrix can be found by the product of F and the intrinsic camera matrix K.This gives the epipolar lines of the images.

$$E = K_2^T * F * K_1 \tag{1.1}$$

1.5 Decomposing E for rotation and translation

We can find the R and T matrices by doing SVD of e and finding the T matrix as the last column of U and R being the product of matrices of SVD and a W matrix.

$$\begin{aligned} &R = UW^TV^T, t = u_3; \quad && R = UW^TV^T, t = -u_3 \\ &R = UWV^T, t = u_3; \quad && R = UWV^T, t = -u_3 \end{aligned}$$

Figure 1.4: Getting R and T

This gives us 4 possible answers and using the condition listed in [2], we can choose the best one, that has the max points in front of the camera, basically the most number of positive Z values. [3]

1.6 Rectification and Drawing Epilines

We do rectification to make the obtained epilines parallel across the image so that the axis is parallel to the baseline of the image.This is done to make searching for depth and disparity much easier.
The epilines tend to meet at a vanishing point at infinity.Doing rectification makes them parallel to the axis of the image.CV2.stereorectifyuncalibrated was used to estimate the H matrix for the images and according using cv2.warpperspective, the images were warped and the epilines are parallel. Modifying the images also affects the F matrix and the E matrix. [4]

1.7 Correspondence

For this , we try to find matching regions between the images,so that gives us a disparty map of the differences of the images that can be used to get the depth of the image.A window of a specific size is used to search along the epilines so that the time taken is less.The window is slide over every point across the images to find matching regions, that can be seen as blocks and this is called block matching.Here SAD is used to do block matching.Using this depth was found for the images. The time taken is much and depends on image size, window size and nature of the images.

[5]

$$SAD(win_l, win_r) = \sum_x \sum_y |I_{win_l}(x, y) - I_{win_r}(x, y)| \tag{1.2}$$

The window with the least SAD is the most similar window.

1.8 Finding Depth

We can find depth using the disparity, that is given by, [5]

$$Depth = \frac{focal_length * baseline}{disparity} \tag{1.3}$$

Additional References

Lecture slides and previous assignment code for SIFT and RANSAC was used for this project.

Chapter 2

OUTPUTS

The outputs for all the matrices, images and maps can be found at `https://drive.google.com/drive/folders/1g0OZvpk5j5qbTepNfcaK92isy16s5c44?usp=sharing`.

Chapter 3

PROBLEMS FACED

The problems faced are:

- The rectified image for the second data set was not warped properly.
- RANSAC was giving random F matrix for each run.This has to be overcome by running the code multiple times till the value is ideal and the outputs in the code.
- The depth map is taking a lot of time to compute and was made faster by reducing size of the image.
- Using SSD for disparity map was taking more time compared to SAD.
- RANSAC randomly failing to converge due to limits of iterations and noise in the pipeline not being filtered.
- The presence of noise in the F matrix and the depth and disparity maps affects the quality of the output.

REFERENCES

[1] GAtech. *Camera Calibration*. `https://www.cc.gatech.edu/classes/AY2016/cs4476_fall/results/proj3/html/sdai30/index.html`. [Online]. 2022.

[2] CMSC733. *RANSAC*. `https://cmsc733.github.io/2022/proj/p3/#ransac`. [Online]. 2022.

[3] Stackoverflow. *Projection Matrix*. `https://stackoverflow.com/questions/16101747/how-can-i-get-the-camera-projection-matrix-out-of-calibratecamera-return-value`. [Online]. 2013.

[4] Opencv. *Epilines*. `https://docs.opencv.org/3.4/da/de9/tutorial_py_epipolar_geometry.html`. [Online]. 2022.

[5] Apar Garg. *Depth*. `https://medium.com/analytics-vidhya/distance-estimation-cf2f2fd709d8`. [Online]. 2022.