Homework 3: Epipolar Geometry

November 5, 2023

Due Date: November 26 by 23:59:59

Introduction

In this project, you will implement **Fundamental Matrix Estimation**, **Camera Calibration** and **Triangulation**. Starter code for data loading, visualization and evaluation have been provided.

1 Fundamental matrix estimation with ground truth matches (30 pts.)

Run the code and visualize the **library** and **lab** image pairs and their matches.

Add your own code to the **fit_fundamental()** function to calculate the fundamental matrix from ground truth matches.

You are required to implement and compare both the **normalized**(using the **normalize_points()** function) and **unnormalized** algorithms. Refer to slides *07 Epipolar Geometry* for the Eight-Point Algorithm. Don't forget to enforce the rank-2 constraint. This can be done by taking the SVD of F, setting the smallest singular value to zero, and recomputing F.

Function **visualize_fundamental()** and **evaluate_fundamental()** can be used to visualize and verify your results. For each algorithm and each image pair, report your visualization and evaluation results.

2 Camera calibration (30 pts.)

For the **lab** image pair, calculate the camera projection matrices by using 2D matches in both views and 3D point coordinates given in *lab_3d.txt* in the data file. Refer to slide *06 Camera Calibration* page for the calibration algorithm.

Once you have computed your projection matrices, you can evaluate them using the **evaluate_points()** function included in the starter code, which will provide you the projected 2D points and residual error.

(Hint: For a quick check to make sure you are on the right track, empirically this residual error should be < 20 and the squared distance of your projected 2-D points from actual 2-D points should be < 4.)

3 Calculate the camera matrices (10 pts.)

Calculate the camera intrinsic and extrinsic matrices **K, R, T** for the **lab** and **library** pairs using the estimated or provided projection matrices. Projection matrices of **library** are already provided in *library1_camera.txt* and *library2_camera.txt.*

4 Triangulation (20 pts.)

For the **lab** and **library** pairs, use linear least squares to triangulate the 3D position of each matching pair of 2D points given the two camera projection matrices (see slide *06 Camera Calibration* for the method). As a sanity check, your triangulated 3D points for the lab pair should match very closely the originally provided 3D points in *lab_3d.txt.* For each pair, use the sample code to calculate and report the residuals between the observed 2D points and the re-projected 3D points in the two images.

5 Fundamental matrix estimation without ground-truth matches (10 pts.)

The provided **house** and **gaudi** image pairs do not include ground truth 2D matches. You should use the SIFT descriptor function to generate keypoints, Brute Force Matcher to match and RANSAC to calculate fundamental matrix. You can use match generation and RANSAC from your putative code in Assignment 2 or Python-OpenCV. For this part, only use the normalized algorithm. Report the number of inliers and the average residual for the inliers, and display the inliers in each image.

6 Report

You have now completed the entire process of this project. Put all the calculation(fundamental and projection matrices, triangulation), visualization and evaluation results in your report. More experiments and discussions are encouraged and may get a bonus.

7 Hints

7.1 Useful Functions

Here is a list of potentially useful functions:

- `np.linalg.svd`
- `scipy.linalg.rq`
- `np.linalg.qr`
- `cv2.SIFT_create`
- `cv2.BFMatcher`
- `Any necessary APIs...`

Here are some supplemental materials:

- SVD Decomposition: https://blog.csdn.net/qq_35987777/article/details/109557291
- QR Decomposition: https://zhuanlan.zhihu.com/p/112327923
- RANSAC: https://blog.csdn.net/gongdiwudu/article/details/112789674
- SIFT: https://zhuanlan.zhihu.com/p/536619540

7.2 How to solve the linear least square with SVD decomposition

Homogeneous linear least square:

$$min\,||Ax|| \tag{1}$$

For homogeneous linear systems, the meaning of a least-squares solution is modified by imposing the constraint:

$$||x|| = 1 \tag{2}$$

Compute the SVD decomposition of A:

$$A = UDV^T \tag{3}$$

$$U^TU = I \tag{4}$$

$$VV^T = I \tag{5}$$

Therefore,

$$||Ax|| = ||UDV^Tx|| = ||DV^Tx|| = ||Dy|| \tag{6}$$

where $y = V^Tx$. The question is equivalent to minimize $||Dy||$ and satisfy $||y|| = 1$. Let $y = (0, ..., 1)$, then $||Ax|| = \sigma_n$, where σ_n is the smallest eigen value.

7.3 Sample of building a homogeneous linear equations

Take camera calibration algorithm as a example:

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Therefore,

$$x_i = \frac{p_{11}X_i + p_{12}Y_i + p_{13}Z_i + p_{14}}{p_{31}X_i + p_{32}Y_i + p_{33}Z_i + p_{34}}$$

$$y_i = \frac{p_{21}X_i + p_{22}Y_i + p_{23}Z_i + p_{24}}{p_{31}X_i + p_{32}Y_i + p_{33}Z_i + p_{34}}$$

Transform them into equations,

$$p_{11}X_i + p_{12}Y_i + p_{13}Z_i + p_{14} - x_i * (p_{31}X_i + p_{32}Y_i + p_{33}Z_i + p_{34}) = 0$$

$$p_{21}X_i + p_{22}Y_i + p_{23}Z_i + p_{24} - y_i * (p_{31}X_i + p_{32}Y_i + p_{33}Z_i + p_{34}) = 0$$

We get a homogeneous linear equations:

$$\begin{pmatrix} X_i & Y_i & Z_i & 1 & 0 & 0 & 0 & 0 & -x_iX_i & -x_iY_i & -x_iZ_i & -x_i \\ 0 & 0 & 0 & 0 & X_i & Y_i & Z_i & 1 & -y_iX_i & -y_iY_i & -y_iZ_i & -y_i \end{pmatrix} \begin{pmatrix} p_{11} \\ P_{12} \\ \cdots \\ p_{33} \\ p_{34} \end{pmatrix} = 0$$