Spring 2023

Project 3 Report

Perception for autonomous robots

April 16 2023

Student:
Rishikesh Jadhav

Instructors:
Dr. Samer Charifa

Course code:
ENPM 673

Contents

1 Problem 1

This problem consists of 5 tasks -

1. The minimum number matching points to solve this mathematically.
2. The pipeline that needs to be done in order to calibrate the camera from the given image.
3. Find the Projection matrix P.
4. From P find,K,R,T
5. Additionally, find Reprojection error.

1.1 Pipeline

Camera calibration is important as it corrects distortions and imperfections, producing accurate and reliable images. The process determines the intrinsic and extrinsic parameters of the camera, enabling accurate measurement of distance, size, and shape. The following is the pipeline for calibrating a camera.

1. Checkerboard images are captured from different angles to gather data for calibration
2. The world coordinates of the checkerboard corners are determined, and the corresponding image coordinates are found using the cv2.findChessboardCorners function.
3. The camera parameters are calculated using the P matrix, which is calculated by solving the equation I = PW, where I is the image point and W is the world point.
4. After calculating P we calculate C which is the left elements of P.
5. RQ factorization is then performed on C to obtain the camera matrix, K, and the rotation matrix, R
6. Extract the Rotation Matrix by multiplying rz,ry,rx.
7. Extract the Translation vector by decomposing the P matrix and taking the last column of v.Transpose.
8. Find Reprojection error for each point.
9. Alternatively, the above steps can also be accomplished using the cv2.calibrateCamera function, which automatically detects and finds the checkerboard corners, calculates the camera matrix, distortion coefficients, rotation, and translation vectors, and provides them as output.

1.2 Results

1. **Minimum number of matching points** The minimum number of matching points required to solve for the intrinsic matrix mathematically is 6. However, it is recommended to have more than 6 points to obtain a more accurate calibration.
2. **Mathematical Formulation**We are given image and world coordinates so we calculate P using the equation (1):

$$\begin{bmatrix} x_{image} & y_{image} & 1 \end{bmatrix} = P \begin{bmatrix} x_{world} & y_{world} & z_{world} & 1 \end{bmatrix} \tag{1}$$

Here, P can be decomposed as given in equation (2):

$$P = K[P \mid \widetilde{C}] \tag{2}$$

To compute P, let's assume P_1^T, P_2^T, and P_3^T are three row vectors of P. We compute A as given in equation (3). Where $AP = 0$.

$$A = \begin{bmatrix} 0^T & -w_i' X_i^T & v_i' X_i^T \\ w_i' X_i^T & 0^T & -u_i' X_i^T \\ -v_i' X_i^T & u_i' X_i^T & 0^T \end{bmatrix} \tag{3}$$

We solve for P by computing the Singular Value Decomposition (SVD) of A. P is the last row of V obtained by SVD of A reshaped to $(3,4)$. We divide P by its last element to get 1 at the last position. C matrix is calculated by computing SVD of P. C is the last row of V matrix obtained by SVD of P reshaped to $(4,1)$. A M matrix is obtained by the dot product of P and $[I_3 \mid -\widetilde{C}]^{-1}$. RQ factorization of M yields calibration matrix K.

3. **Intrinsic Matrix K** The following is the intrinsic matrix as an output of the above steps and from given image and world points.

$$K = \begin{bmatrix} -6.7912331e+01 & -7.9392768e-02 & 3.3562042e+01 \\ 0 & 6.7619034e+01 & 2.5845427e+01 \\ 0 & 0 & 4.1946620e-02 \end{bmatrix} \tag{1}$$

4. **Projection matrix P**

   Projection matrix:

$$\begin{bmatrix} 2.87364445e+01 & -1.75735415e+00 & -7.00687538e+01 & 7.56890519e+02 \\ -2.01369011e+01 & 6.58890120e+01 & -2.22140404e+01 & 2.13263797e+02 \\ -2.77042391e-02 & -2.59559759e-03 & -3.13888009e-02 & 1.00000000e+00 \end{bmatrix}$$

5. **Rotation matrix R**

   Rotation Matrix:

$$\begin{bmatrix} -0.74948643 & 0.11452983 & -0.65203758 \\ 0.0453559 & 0.99149078 & 0.12202001 \\ 0.66046418 & 0.06187859 & -0.74830349 \end{bmatrix}$$

6. **Translational vector T**

   Translation vector:

$$\begin{bmatrix} 0.64862355 \\ 0.30183152 \\ 0.69751919 \\ 0.04064735 \end{bmatrix}$$

7. **Reprojection errors**

   Reprojection errors:

$$[0.28561276758292053, 0.9725828450556875, 1.0360817843232188, 0.4540862872790896,$$

0.1908983186303205, 0.31899208342793967, 0.19594240508863442, 0.3082960281429625]

2 Problem 2

The objective of this question is to perform camera calibration using the concepts learned in class. The camera calibration process involves estimating the intrinsic camera parameters, such as the camera matrix and distortion coefficients, using a calibration target (checkerboard in this case) and a set of images taken by the camera. The calibration target used is a printed checkerboard with known dimensions of each square.

Overall, the goal of this question is to gain practical experience in camera calibration using real-world images and learn how to use OpenCV functions for corner detection and calibration.

2.1 Pipeline

1. Read in the calibration images from the folder using the os module.
2. Grayscale the images.
3. resize the grayscaled images.
4. Find the corners of the checkerboard pattern in the resized image using the cv2.findChessboardCorners() method.
5. Draw the corners found on the image using the cv2.drawChessboardCorners().
6. Calibrate the camera using the cv2.calibrateCamera() method on the list of corner points to obtain the intrinsic camera parameters.
7. Compute the reprojection error for each image using the cv2.projectPoints() method and the intrinsic camera parameters obtained from the previous step.
8. Extract the camera matrix from the intrinsic camera parameters.

To increase the accuracy of the Camera calibration matrix(K) there are many ways, which are stated as follows:

- Use a larger calibration board
- Use more calibration images
- Capture images from different positions and orientations
- Use accurate measurements for square size.
- Ensure flat and level calibration board

2.2 Results

A few examples of the corners detected are as follows:

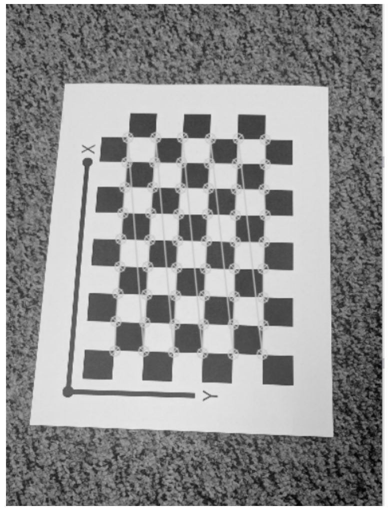

Figure 1: Corners detected example.1

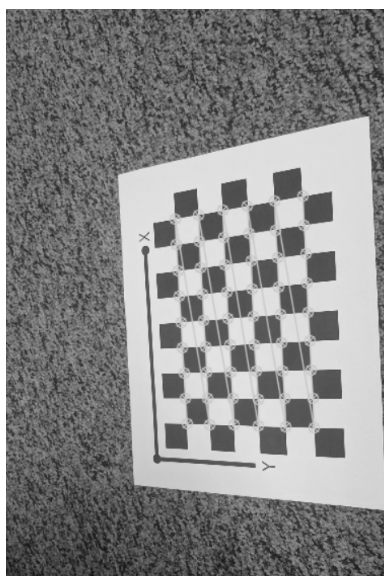

Figure 2: Corners detected example.2

Following are the results which include the reprojection errors fr all the images along with the final Intrinsic camera matrix (K):

1. Reprojection error for image no.0 : 0.119819
2. Reprojection error for image no.1 : 0.261013
3. Reprojection error for image no.2 : 0.409439
4. Reprojection error for image no.3 : 0.541758
5. Reprojection error for image no.4 : 0.221944
6. Reprojection error for image no.5 : 0.353663

7. Reprojection error for image no.6 : 0.051968
8. Reprojection error for image no.7 : 0.224669
9. Reprojection error for image no.8 : 0.481034
10. Reprojection error for image no.9 : 0.404181
11. Reprojection error for image no.10 : 0.481047
12. Reprojection error for image no.11 : 0.513687
13. Reprojection error for image no.12 : 0.429745

$$K = \begin{bmatrix} 2.23165807e+03 & 0.00000000e+00 & 7.78116041e+02 \\ 0.00000000e+00 & 2.45420808e+03 & 1.32346726e+03 \\ 0.00000000e+00 & 0.00000000e+00 & 1.00000000e+00 \end{bmatrix} \tag{2}$$

3 Problems Encountered and solutions

Following road blocks were found during the entire process of completing the project:

1. 1 - In the first question finding the correct K matrix was tricky as the answers were different while the pipeline was correct.
2. 2 - I had to put a condition to make the values which were very low in the k matrix to be equal to zero to ensure having an upper triangular matrix.
3. 3 - The second question was fairly straight forward.

4 References

1. https://leohart.wordpress.com/2010/07/23/rq-decomposition-from-qr-decomposition/
2. https://www.cc.gatech.edu/classes/AY2016/cs4476$_fall/results/proj3/html/agartia3/index.htmlhttps://docs.opencv.org/2.4/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html$

3. https://medium.com/analytics-vidhya/camera-calibration-with-opencv-f324679c6eb7