CS-512 Assignment 5: Report

Russi Sinha
Department of Computer Science
Illinois Institute of Technology

Abstract

This is a report for Assignment 5 for CS512. The task given at hand is to implement a camera calibration algorithm under the assumption of noisy data using planar or non-coplanar calibration and robust estimation using RANSAC to eliminate outliers.

1. Problem Statement

Implement a camera calibration algorithm using planar or non-coplanar calibration and remove outliers using RANSAC.

2. Proposed Solution

For solving this problem statement planar calibration has been used for camera calibration and RANSAC is used to eliminate outliers

3. Implementation Details

For this execution a simple chessboard has been used which is converted to grayscale and the corners are found using the OpenCV function “findChessboardCorners”. If corners are found, they are improved by using “cornerSubPix” and the points are displayed on the image using the function “drawChessboardCorners”. These 2D image points along with the corresponding 3D points are saved in a text file.

To calculate the parameters, load the points from the text file and split them into image and world points. Create a matrix ‘M’ with 2m*12 rows where (m being the number of 3D-2D coordinate pairs in the list) with each set of 2 rows having the following format:
Row1 = [X, Y, Z, 1, 0, 0, 0, 0, -X*u, -Y*u, -Z*u, -u]
Row2 = [0, 0, 0, 0, X, Y, Z, 1, -X*v, -Y*v, -Z*v, -v]

Applying SVD on the matrix we get the values of u, sigma, v_transpose.
Using these we generate the vectors a1, a2, a3 and b.

Since these parameters are known we can calculate the unknown parameters using the following formulas:

$$
\begin{aligned}
|\rho| &= 1/|a_3| \\
u_0 &= |\rho|^2 a_1 \cdot a_3 \\
v_0 &= |\rho|^2 a_2 \cdot a_3 \\
\alpha_v &= \sqrt{|\rho|^2 a_2 \cdot a_2 - v_0^2} \\
s &= |\rho|^4 / \alpha_v (a_1 \times a_3) \cdot (a_2 \times a_3) \\
\alpha_u &= \sqrt{|\rho|^2 a_1 \cdot a_1 - s^2 - u_0^2} \\
K^* &= \begin{bmatrix} \alpha_u & s & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}
\end{aligned}
$$

$$
\begin{aligned}
T^* &= |\rho|(K^*)^{-1} b \\
r_3 &= |\rho| a_3 \\
r_1 &= |\rho|^2 / \alpha_v a_2 \times a_3 \\
r_2 &= r_3 \times r_1 \\
R^* &= [r_1^T \;\; r_2^T \;\; r_3^T]^T
\end{aligned}
$$

The values retrieved by the above formulas are provided in the next section.

4. Results and Discussion

Original image:

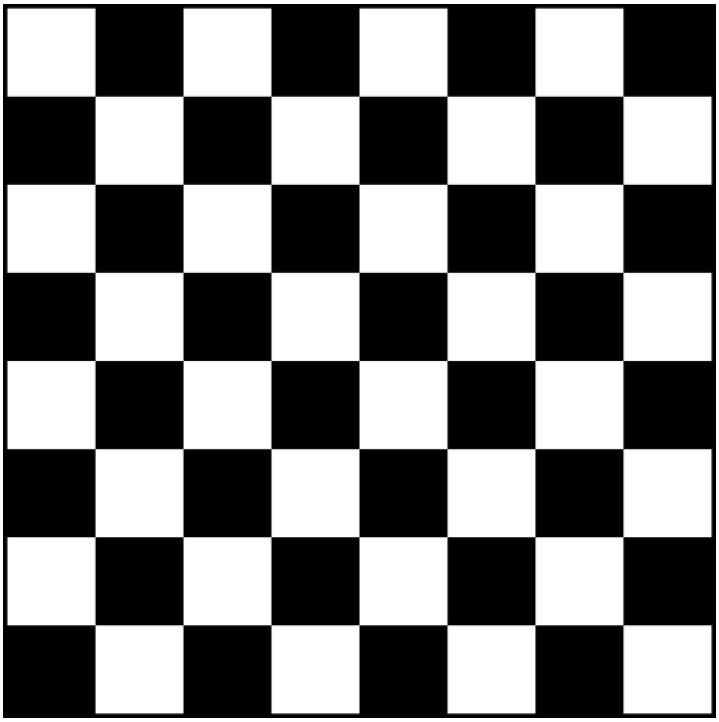
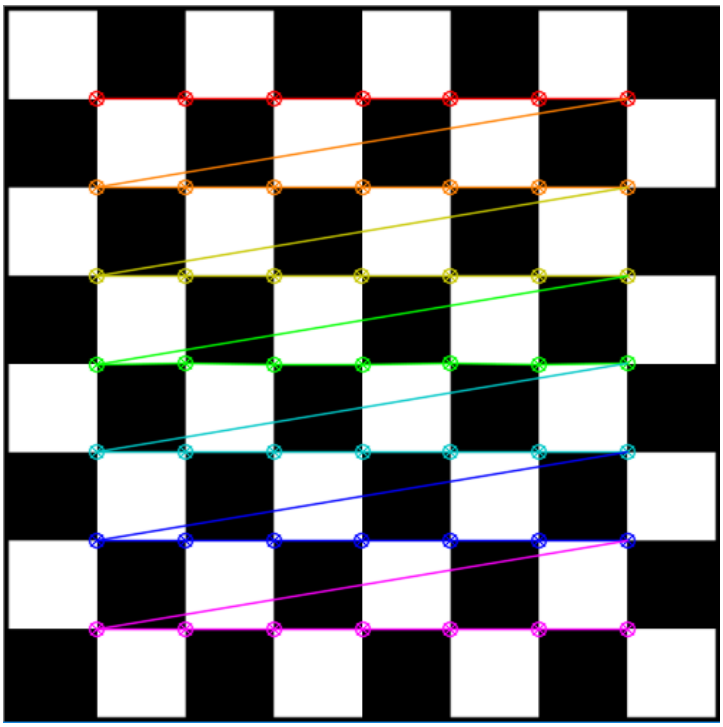


Image with detected corners:



Parameters determined by the program using test data:

u0 = 320.000170391240
v0 = 239.99997093086554
alpha_u = 652.1740688313259
alpha_v = 652.1740748020753
s = -3.39846659375858e-05
K* = [6.52174069e+02, -3.39846659e-05, 3.20000170e+02],
[0.00000000e+00, 6.52174075e+02, 2.39999971e+02],
[0.00000000e+00, 0.00000000e+00, 1.00000000e+00]

T* = [-2.57721739e-04, 3.27185104e-05, 1.04880905e+03]

R* = [-7.68221190e-01, 6.40184508e-01, 1.46353678e-07],
[4.27274298e-01, 5.12729182e-01, -7.44678091e-01],
[-4.76731452e-01, -5.72077427e-01, -6.67423808e-01]

These values are almost equivalent to the known parameters provided along with the test coordinates

Executing the program on the custom data the following results were achieved:

u0 = 7.49335547109877e-17
v0 = 1.7481431988182056e-15
alpha_u = 1.3725936867762369e-17
alpha_v = 6.847795927865924e-17
s = -6.5738374218599945e-18
K* = [1.37259369e-17, -6.57383742e-18, 7.49335547e-17],
     [0.00000000e+00, 6.84779593e-17, 1.74814320e-15],
     [0.00000000e+00, 0.00000000e+00, 1.00000000e+00]

T* = [7.98226216e-01, -6.32054335e-02, -2.22155716e-18]

R* = [6.49602353e-01, 5.37595007e-01, 5.37595007e-01],
     [7.60274150e-01, -4.59338229e-01, -4.59338229e-01],
     [-3.02123688e-17, 7.07106781e-01, -7.07106781e-01]

5. References

https://docs.opencv.org/2.4/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html
https://docs.opencv.org/3.4.3/dc/dbb/tutorial_py_calibration.html