CS512 – AS5 - Report

Akshay Jain

Department of Computer Science

Illinois Institute of Technology

April 26, 2022

Abstract

In this assignment, we have implemented Camera Calibration and Review Questions.The camera calibration aims to determine the geometric parameters of the image formation process. This is a crucial step in many computer vision applications especially when metric information about the scene is required. In these applications, the camera is generally modeled with a set of intrinsic parameters (focal length, principal point, skew of axis) and its orientation is expressed by extrinsic parameters (rotation and translation). Both intrinsic and extrinsic parameters are estimated by linear or nonlinear methods using known points in the real world and their projections in the image plane. These points are generally presented as a calibration pattern with known geometry, usually a flat chessboard.

1. Problem Statements

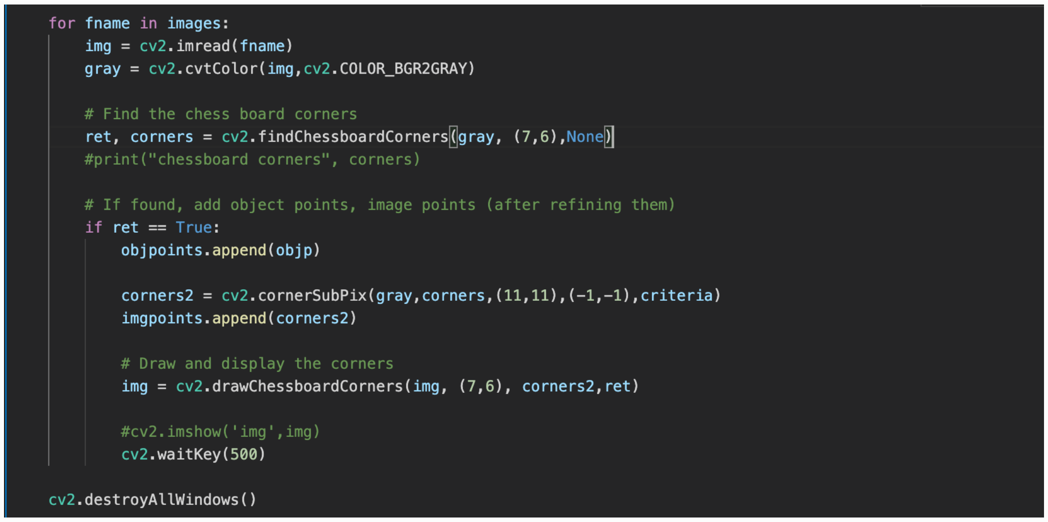
We were asked to create a program that will take images and give us 3d object points as well as 2d image points. Using this generated file we needed to find the cameras intrinsic and extrinsic parameters and based on the values we needed to find the mean square error.

2. Proposed solution

1) Feature Point Extraction

Using ***cv2.findChessboardCorners()*** and ***cv2.cornerSubPix()*** we found out the corner points in the given chess board image. Using

cv2.drawChessboardCorners() we draw the pattern on the image.



```
for fname in images:
    img = cv2.imread(fname)
    gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)

    # Find the chess board corners
    ret, corners = cv2.findChessboardCorners(gray, (7,6),None)
    #print("chessboard corners", corners)

    # If found, add object points, image points (after refining them)
    if ret == True:
        objpoints.append(objp)

        corners2 = cv2.cornerSubPix(gray,corners,(11,11),(-1,-1),criteria)
        imgpoints.append(corners2)

        # Draw and display the corners
        img = cv2.drawChessboardCorners(img, (7,6), corners2,ret)

        #cv2.imshow('img',img)
        cv2.waitKey(500)

cv2.destroyAllWindows()
```

For this program we gave 3 chessboard images and generated correspondence points file for each of them and stored in a text file. This file contained 3d object point to their corresponding 2d image point in them.

2) Calibration

We read the text file in numpy array and separated world 3d point and 2d image points and created a homography matrix with respect to each file. This homography matrix contains rows as Pstar, zeros, -x*Pstar and zeros,Pstar, -y*Pstar corresponding to each pair of corresponding points in the file. Using SVD decomposition we found Vtranpose for all the homography matrix.

```
# for img1
u,d,v_t = np.linalg.svd(M)
v = np.transpose(v_t)
v
```

I took transpose of this and used last column to get the value of h1,h2 and h3 for Hhat matrix. After forming Hhat matrix for all the files I used below function named *fnvij* to get V11,V12 and V22 for each matrix and formed a matrx.

```
#Vij for img1
def fnvij(h,i,j):
   h=h.T
   a = np.array([h[i-1][0]*h[j-1][0], h[i-1][0]*h[j-1][1] + h[i-1][1]*h[j-1][0],
h[i-1][1]*h[j-1][1], h[i-1][2]*h[j-1][0]+ h[i-1][0]*h[j-1][2],
    h[i-1][2]*h[j-1][1]+ h[i-1][1]*h[j-1][2], h[i-1][2]*h[j-1][2]])
   return a
```

Taking this matrix SVD decomposition and Using last column of the V matrix I found S11, S12, S22, S13, S23 and S33. By using these values I found out c1, c2, v_, lambda, au, av, s and u0.

```
#parameters
c1 = s11*s13 -s11*s23
c2 = s11*s22 -s12*s12
v_ =c1/c2
lambdha = s33 - (s13*s13 + v_*c1)/s11
au =np.sqrt(lambdha/s11)
av = np.sqrt(lambdha/c2)
s = -(s12*(au*au)*av)/lambdha
u0 = (s*v_/au) - (s13*(au*au)/lambdha)
```

By considering these values I found the intrinsic and extrinsic parameter of the camera naming Rstar and Tstar.

```
K_star = [[au, s, u0], [0, av, v_], [0, 0, 1]]

K_star_inv = np.linalg.inv(K_star)

T_star = alpha * K_star_inv* h3

r1 = alpha* K_star_inv* h1
r2 = alpha* K_star_inv* h2
r3 = np.cross(r1, r2)
```

Input chessboard image:

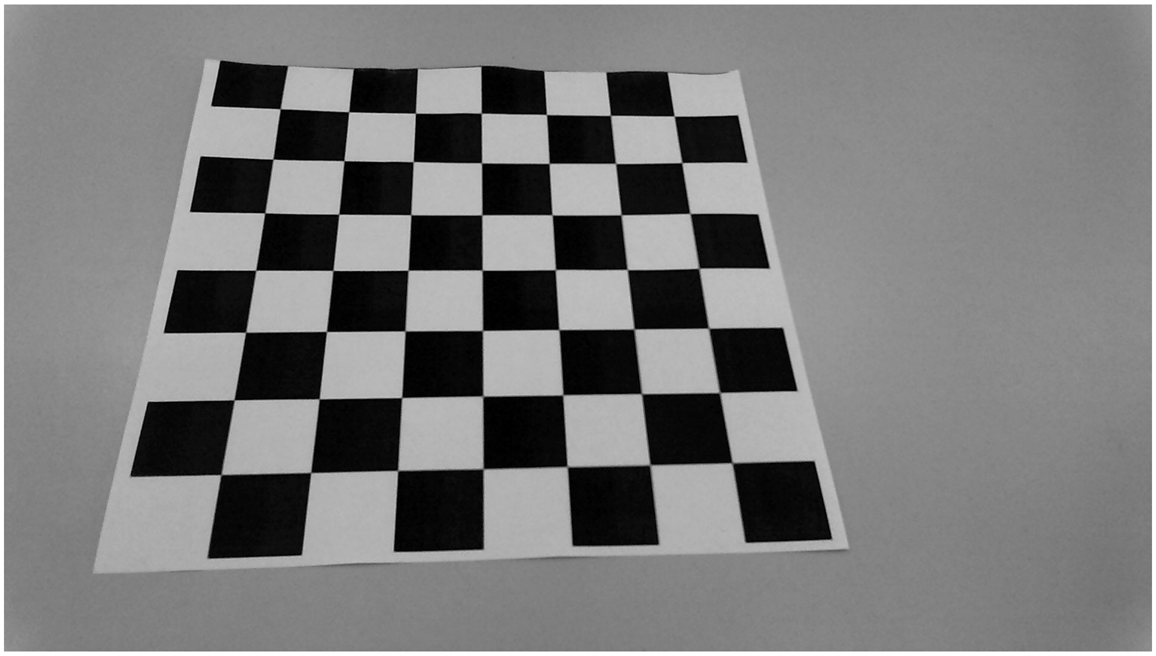
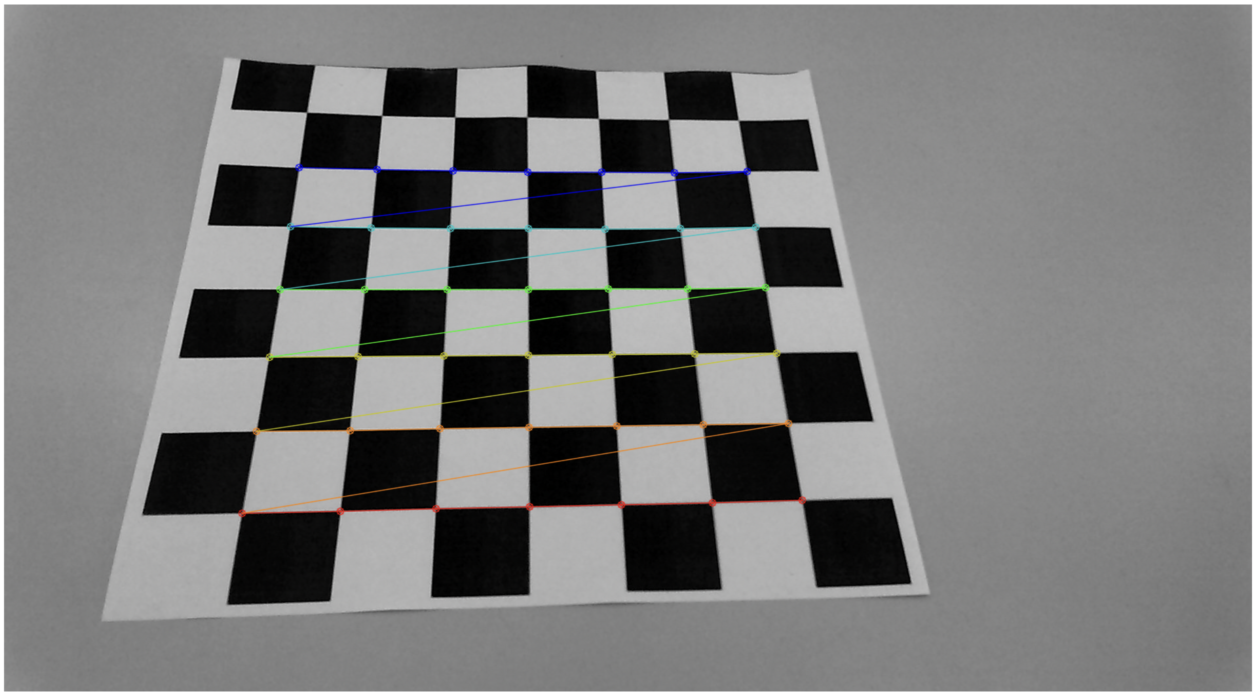
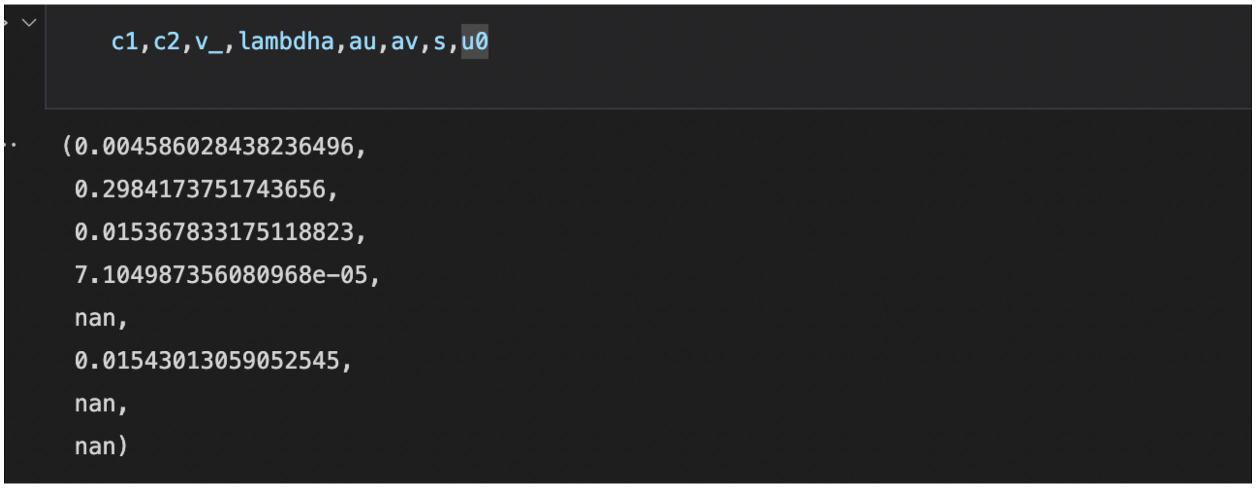


Fig showing Chessboard Corners after finding the 2d and 3d corresponding points



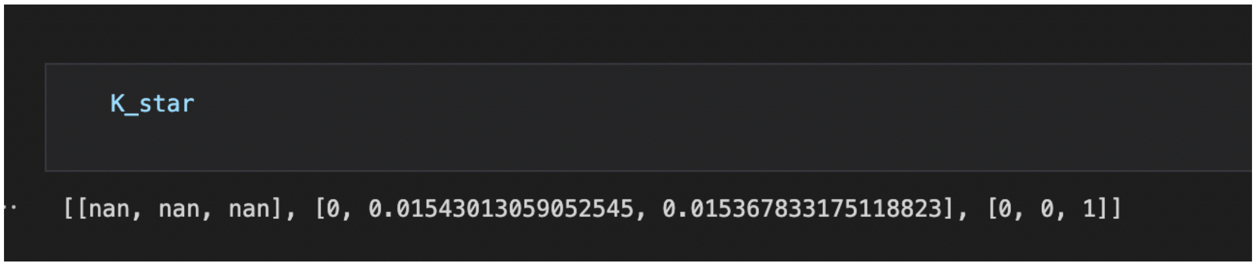
5. Results comparison and Conclusion:

Below are the c1, c2, v0, lambda, alphaU, alphaV, s and u0 values



```
c1,c2,v_,lambdha,au,av,s,u0
```

```
(0.004586028438236496,
 0.2984173751743656,
 0.015367833175118823,
 7.104987356080968e-05,
 nan,
 0.01543013059052545,
 nan,
 nan)
```



```
K_star
```

```
[[nan, nan, nan], [0, 0.01543013059052545, 0.015367833175118823], [0, 0, 1]]
```

6. References: -

- https://opencv24-python-tutorials.readthedocs.io/en/stable/py_tutorials/py_calib3d/py_calibration/py_calibration.html
- https://numpy.org/doc/