Computer Vision CS534

Homework 3:

Camera Calibration and Augmented Reality

Hao Luo

RUID: 162006265

Part I: Camera Calibration using 3D calibration object

Draw the image points, using small circles for each image point.

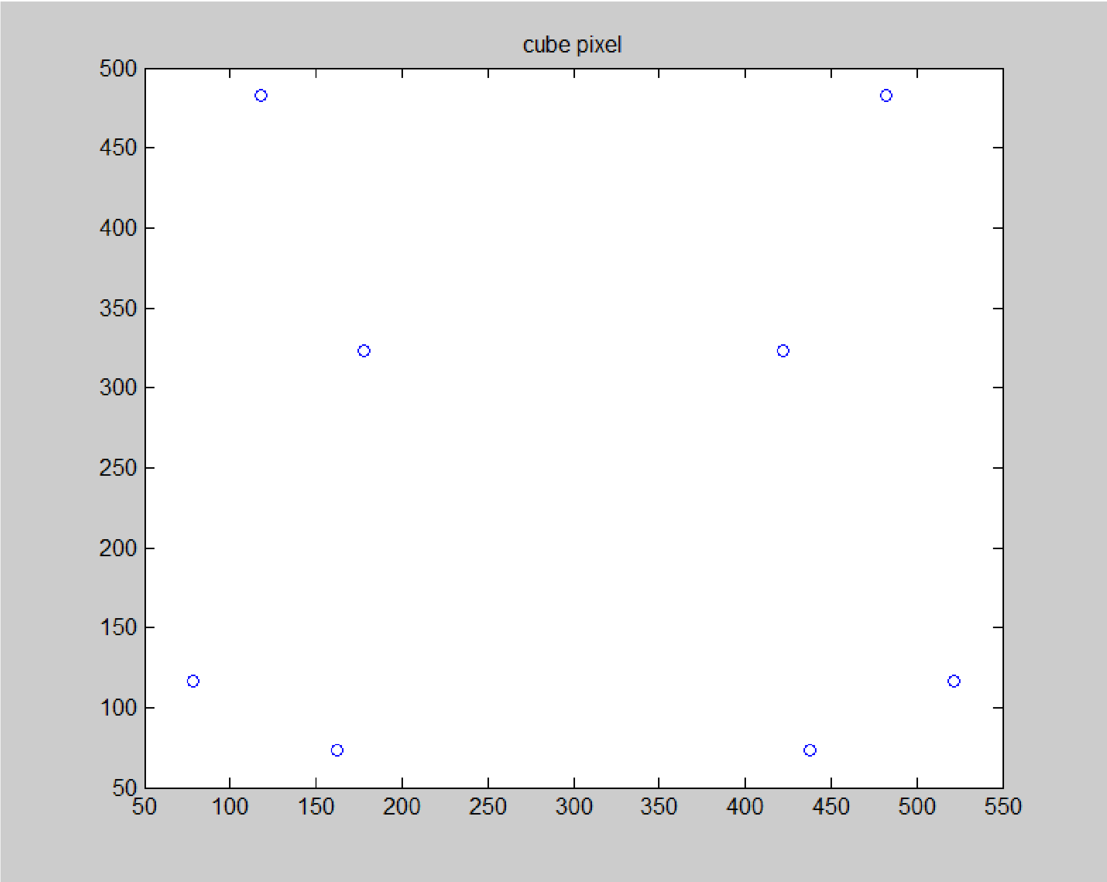


Figure1: 8 image points

The matrix P is:

2	2	2	1	0	0	0	0	-844	-844	-844	-422
0	0	0	0	2	2	2	1	-646	-646	-646	-323
-2	2	2	1	0	0	0	0	356	-356	-356	-178
0	0	0	0	-2	2	2	1	646	-646	-646	-323
-2	2	-2	1	0	0	0	0	236	-236	236	-118
0	0	0	0	2	2	-2	1	966	-966	966	-483
2	-2	2	1	0	0	0	0	-964	-964	964	-482
0	0	0	0	2	-2	2	1	-966	-966	966	-483
-2	-2	-2	1	0	0	0	0	-876	876	-876	-438
0	0	0	0	-2	-2	-2	1	-146	146	-146	-73
2	-2	-2	1	0	0	0	0	324	324	-324	-162
0	0	0	0	2	-2	-2	1	146	146	-146	-73
-2	-2	-2	1	0	0	0	0	156	156	156	-78
0	0	0	0	-2	-2	-2	1	234	234	234	-117
2	-2	-2	1	0	0	0	0	-1044	1044	1044	-522
0	0	0	0	2	-2	-2	1	-234	234	234	-117

The matrix M is:

0.1925	0.0283	0.0786	0.7346
0.0000	0.2044	0.0001	0.6120
0.0000	0.0001	0.0003	0.0024

The corresponding 3 Euclidean coordinates of the camera center in the frame of reference of the cube

center =

-0.0000

-2.9912

-8.2695

The M' is:

734.6289	107.8955	299.9999
0.0009	780.1442	0.2641
0.0000	0.3597	1.0000

θ_x is:

$$\theta_x = -0.3452 * \mathrm{pi}$$

Matrix Rx is:

1.0000	0	0
0	0.9410	0.3384
0	-0.3384	0.9410

Matrix N is

734.6289	-0.0000	318.8125
0.0000	734.0199	264.2723
0.0000	0.0000	1.0627

θ_z is:

$$\theta_z = (-1.2602e-06)*\mathrm{pi}$$

Matrix Rz is

1.0000	0.0000	0
-0.0000	1.0000	0
0	0	1.0000

We can directly compute the calibration matrix K by using K=N*Rz;

K is

691.2797	0.0009	299.9999
0.0000	690.7067	248.6780
0.0000	0.0000	1.0000

So, the focal lengths is $\alpha = 691.2797$, $\beta = 690.7067$;

The pixel coordinates of the image center of camera is

(299.9999, 248.6780)

Part 2: Camera Calibration using 2D calibration object

1. Corner Extraction and Homography computation

In this part, I first compute the 3d coordinate of each corner of grid.
As there is 9 squares in width and 7 in height (each square is 30mm*30mm), I use A(0 0), B(0,210), C(270, 210) and D(270 0) to represent the 3d coordinates.
In order to correspond with the 3d coordinates ,I recommend the user to select the up left corner first, then bottom left , bottom right and finally up right corner when using the ginput function to manually select the grid corner.

And we get the homography H of each picture:

homography H of images2

0.9676	0.0846	36.4537
0.0233	-0.8766	228.6640
0.0000	0.0002	0.5523

homography H of images9

1.0768	0.0409	64.4815
0.1405	-0.9314	208.2112
0.0005	0.0002	0.4922

homography H of images12

-0.7132	-0.0497	-63.5125
0.1719	0.8973	-248.3903

0.0005	-0.0002	-0.6288

homography H of images20

-0.8558	-0.2754	-65.0941
0.0141	0.4047	-143.0020
0.0000	-0.0008	-0.5126

2. Computing the parameters

matrix B is

-0.0000	0.0000	0.0005
0.0000	-0.0000	0.0004
0.0005	0.0004	-1.0000

lambda =-0.7841

alpha =727.2152

beta =712.2538

gamma =1.9064

v0 =217.5892

u0 =-127.0573

K =

727.2152	1.9064	-127.0573
0	712.2538	217.5892
0	0	1.0000

Then we can get t and R:

t1 =

109.6207

114.1784

414.0507

R1 =

0.9997 0.1209 0.0123

0.0219 -0.9765 -0.1776

0.0086 0.1787 -0.9788

R1T*R1=

1.0000	0.1010	0
0.1010	1.0000	0
0	0	0.9898

We can see from the results that RTR is not an identity.

t2 =

116.8326
95.1537
329.9416

R2 =

0.9548	0.0641	0.2967
0.0300	-0.9911	-0.0924
0.2959	0.1166	-0.9482

R2T*R2=

1.0000	0.0660	0.0000
0.0660	1.0000	0.0000
0.0000	0.0000	0.9956

We can see from the results that RTR is not an identity.

t3 =

-168.5206
-134.1289
-538.4894

R3 =

-0.8520	-0.0773	-0.5220
0.0736	0.9880	-0.1541
0.5184	-0.1338	-0.8361

R3T*R3=

1.0000	0.0693	0.0000
0.0693	1.0000	0.0000
0.0000	0.0000	0.9952

We can see from the results that RTR is not an identity.

t4 =

-145.1786
-35.8529
-415.8293

R4 =

-0.9998	-0.4078	-0.0173
0.0125	0.6377	-0.6593
0.0144	-0.6535	-0.6325

R4T*R4=

1.0000	0.4062	-0.0000
0.4062	1.0000	0
-0.0000	0	0.8350

We can see from the results that RTR is not an identity.
Then we compute the new R and RTR after enforcing the rotation matrix constraints, the new RTR is an identity matrix now.

R1_new =

0.9974	0.0705	0.0124
0.0716	-0.9813	-0.1785
-0.0004	0.1789	-0.9839

(R1_new)T*R1_new=

1.0000	-0.0000	-0.0000
-0.0000	1.0000	0
-0.0000	0	1.0000

R2_new =

0.9542	0.0326	0.2974
0.0628	-0.9937	-0.0926
0.2925	0.1070	-0.9503

(R2_new)T*R2_new=

1.0000	-0.0000	-0.0000
-0.0000	1.0000	-0.0000
-0.0000	-0.0000	1.0000

R3_new =

-0.8508	-0.0479	-0.5233
0.0394	0.9872	-0.1544
0.5240	-0.1520	-0.8381

(R3_new)T*R3_new=

1.0000	-0.0000	0.0000
-0.0000	1.0000	-0.0000
0.0000	-0.0000	1.0000

R4_new =

$$\begin{matrix} -0.9777 & -0.2093 & -0.0190 \\ -0.1315 & 0.6798 & -0.7215 \\ 0.1639 & -0.7029 & -0.6922 \end{matrix}$$

(R4_new)T*R4_new=

$$\begin{matrix} 1.0000 & -0.0000 & -0.0000 \\ -0.0000 & 1.0000 & 0 \\ -0.0000 & 0 & 1.0000 \end{matrix}$$

2. Improving accuracy

Figure with the image and approximate grid locations

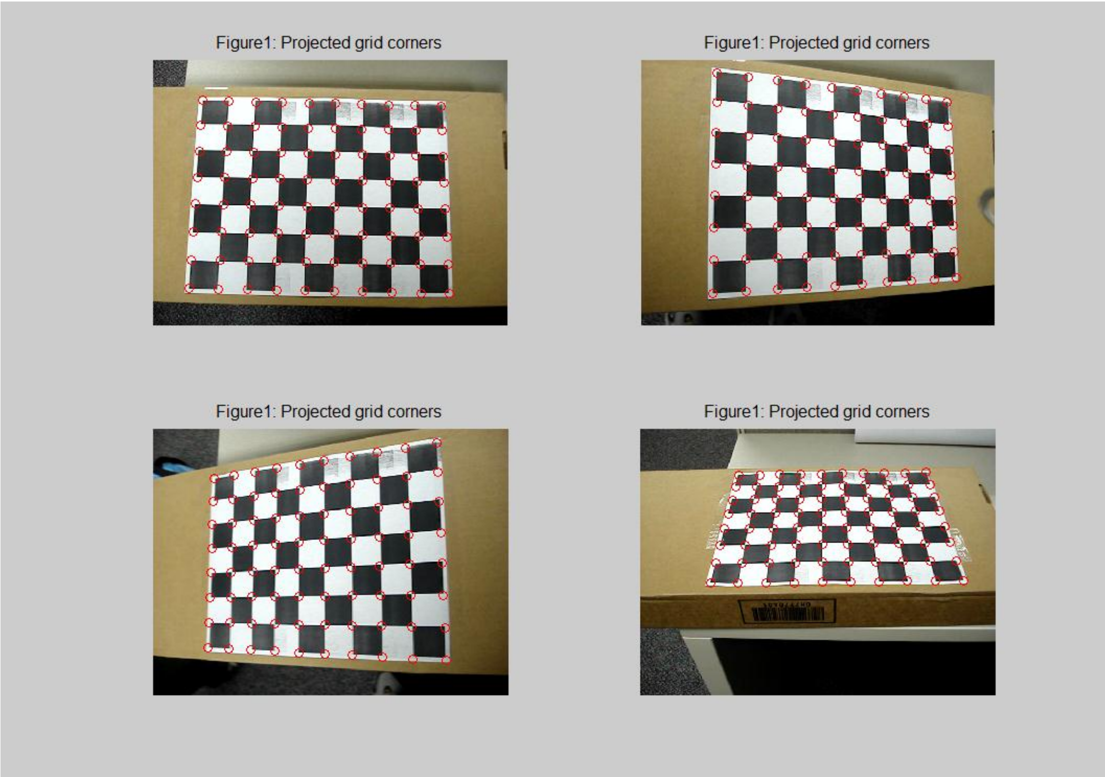


Figure with overlayed Harris corners and correct corners based on it:

Image2

Image9

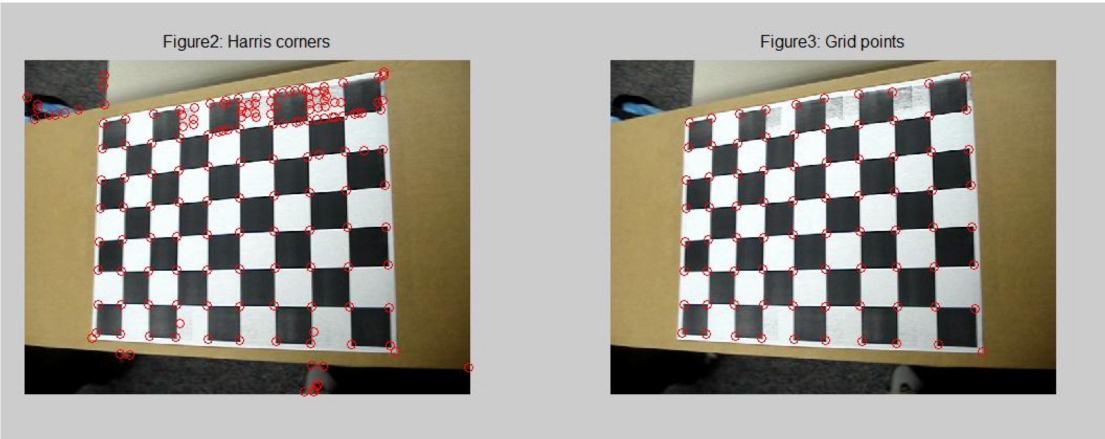


Image12

H1_correct =

-0.9702	-0.0884	-35.1004
-0.0175	0.8906	-230.1412
-0.0000	-0.0002	-0.5554

H2_correct =

-1.0992	-0.0389	-63.0858
-0.1489	0.9467	-208.3384
-0.0005	-0.0001	-0.4906

H3_correct =

0.7110	0.0548	63.7943

-0.1793	-0.9003	248.9024
-0.0005	0.0002	0.6306

H4_correct =

0.8525	0.2662	63.4735
-0.0069	-0.4022	139.6967
0.0000	0.0008	0.5041

I used the max standard deviation of each point to measure the errors:

error =(30.0250, 37.3452, 32.3292, 24.3814)

error_reprojection =(18.7389, 19.7658, 20.9566, 17.7024)

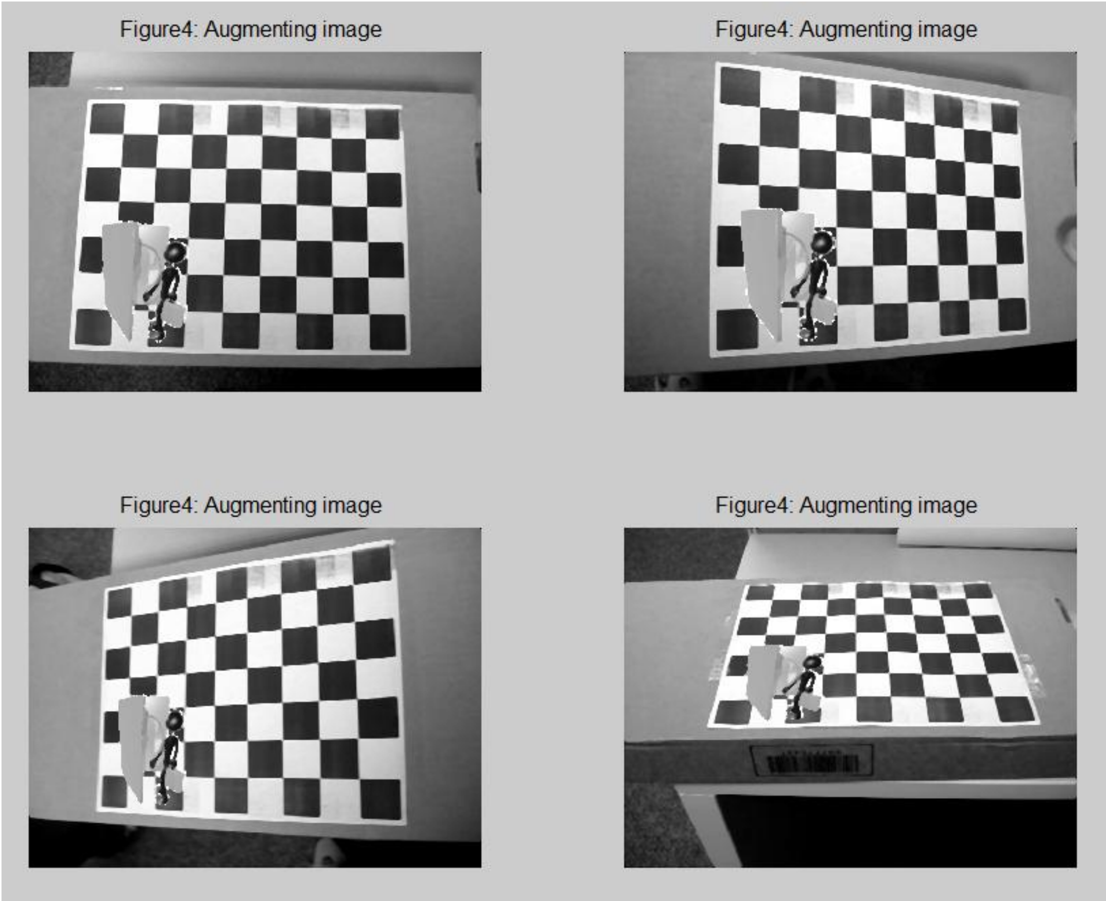
Can you suggest a way this can be done automatically ?

Like the harris corner detection algorithm in figure 2, we can find some unexpected 'wrong corners' have been detected.(something like shoes,labels, desks or other things)
If we can pick those image in which corners are different from the surroundings, the computer can automatically do the work well.

Part 3: Augmented Reality

1. Augmenting an Image

My last 4 digits of ruid id is 6265 ,so I use the clip art 9.jpg



2.Augmenting an Object

According to the function

$$\mathbf{s}\begin{bmatrix}u\\v\\1\end{bmatrix} = K[r_1 \quad r_2 \quad t]\begin{bmatrix}X\\Y\\1\end{bmatrix}$$

We can get the projected coordinates of the cube.

3D coordinates of the cube in img2

66.4933	76.9045	18.1387	2.2326	221.4539	226.6530	207.5872	200.1009
411.8972	260.3053	237.7810	430.3481	415.1821	263.4712	241.7846	434.5440
1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

3D coordinates of the cube in img9

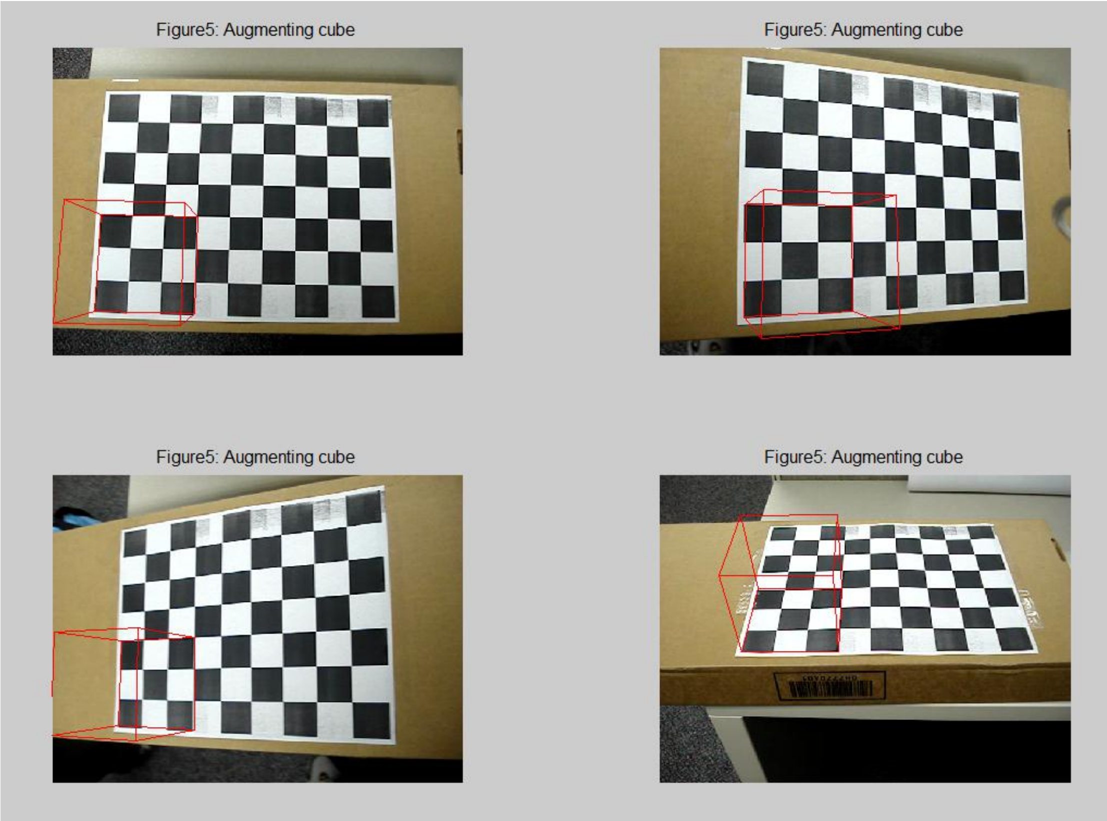
131.7450	134.7215	161.2083	158.3055	301.7191	300.0421	369.1625	373.6843
420.5055	243.5551	222.0356	453.8594	411.1262	248.0011	229.8013	438.4544
1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

3D coordinates of the cube in img12

103.7423	107.8111	4.1333	-3.8804	221.3930	222.7370	134.5978	130.0474
392.1058	259.3605	246.2538	406.5547	397.7039	254.0585	238.4598	414.9328
1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

3D coordinates of the cube in img20

127.9889	153.2371	126.8169	92.3329	279.2316	284.5410	277.9184	270.4576
275.3404	178.5522	64.5261	158.1248	275.3376	177.9516	63.0222	157.1360
1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000



Part 4:Extra credit

With the limitation of 2 images of grid,

$$\mathrm{V} = \begin{bmatrix} v_{12}^T \\ (v_{11} - v_{22})^T \end{bmatrix}$$

V has 4 rows which means we can drive at most 4 parameters from B.
if we define the intrinsic parameter $\gamma=0$, the left 4 intrinsic parameters can be computed.
And then the same way to get the extrinsic parameters.