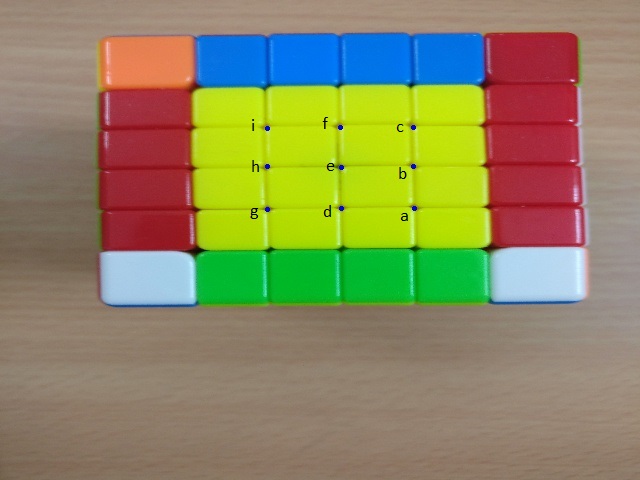
Reflection Essay

# Focus

In this task we need to find the camera parameters for our mobile phone camera. For this task we used the following two images with 9 points in each image as shown below.





The world coordinates are as follows:

a – (0,0,0), b - (1,0,0), c – (2,0,0),

d – (0,1,0), e – (1,1,0), f – (2,1,0),

g – (0,2,0), h – (1,2,0) , i – (2,2,0)

The images are first resized to (640, 480).

The pixel coordinates are hardcoded in the code. They were found by using the mouse button down event (cv2.EVENT\_LBUTTONDOWN) of opencv.

Then cv2.calibrateCamera() was called with the world and pixel coordinates to get the camera marix.

From the matrix we found the focal lengths fx and fy in pixel units. To convert them into mm, we need the actual sensor size in mm.

We found from <https://www.devicespecifications.com/en/model/446c3f91> that the sensor size is 5.22\*3.92.

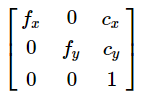
We have (http://ksimek.github.io/2013/08/13/intrinsic/):

Where Fx , Fy are focal length along the x and y axis respectively in mm, fx , fy are focal length along the x and y axis respectively in pixels, w and h are image width and height and W and H are sensor width and height in mm.

Using the above formulas we found Fx = 25.58 mm and Fy = 23.52 mm. The actual focal length is 25 mm . The difference can be attributed to the noise in the real world and pixel coordinates. Also the actual pixel coordinates will be floating values but opencv return only interger pixel values with cv2.EVENT\_LBUTTONDOWN hence the pixel coordinates are approximate integer values which adds to the noise.

Q: What are the returned values of the function cv2. calibrateCamera() ?

A: The first object it returns is the RMS re-projection error. The second thing it return is the camera matrix given by:

 where fx and fy are scaled focal lengths in pixel along x and y axis respectively and cx and cy are the principal point offsets.

It also return the distortion coefficients, and the camera rotation and translation vectors.

Q: Why is the original image resolution not needed?

A: The camera matrix is calculate in the scale of the given pixel coordinates. So it doesn’t matter what the original resolution was as long as the pixel coordinates are calculated in the current resolution. Also while calculating the focal length, we divide by the width and height of the image so the focal length in real world coordinates is independent of resolution of the image as it should be.

Q: Report your observations such as the number of focal lengths, discrepancies, and so on.

A: This has been answered above.

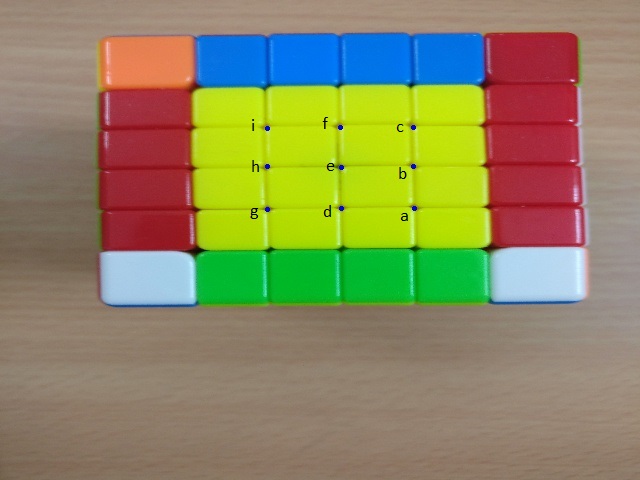
# Reproject

In this task we have to find the reprojection error by first finding the camera intrinsic and extrinsic parameter and using those to reproject the world coordinates to pixel coordinates and finding l2 norm distance from the ground truth.

We chose the below 3 images and the corresponding points.







The world coordinates are the same as the previous question.

Using them we find the camera intrinsic and extrinsic matrix.

After that we use cv2. projectPoints() to project the points of the images and calculate the l2 normed distance of the original and projected pixel points.

The reprojection error comes out to be:- 0.2217

Output:



Q: Comment on the role of the number of images

A: The more the images and more the points, the more robust the output is since it’s a optimization algorithm. It will be less susceptible to noise in the coordinates with more images.

# I wish I had that mobile phone!

We are given co-ordinates of six points in two different images and are asked to find the camera parameters. We have try two different methods with image size as 640x480.

**Method 1:** We considered the world coordinates of these 6 points as [[0, 0, 0], [0, 3, 0], [3, 0, 0], [3, 3, 0], [6, 0, 0], [6, 3, 0]]. We got camera matrix and re-projection error as:

K = [[7.73485965e+03 0.00000000e+00 3.43376325e+02]

[0.00000000e+00 5.52342227e+03 2.33074769e+02]

[0.00000000e+00 0.00000000e+00 1.00000000e+00]]

Reconstruction Error: 0.14585723595027672

**Method 2:** We considered the world coordinates of these 6 points as one of the image co-ordinates itself. That is we have assumed that origin in real world matches with centre of the camera while clicking that particular image.

When 1st image co-ordinates (first 6 lines of points.txt) are considerd as world co-ordinates, we got camera matrix and re-projection error as:

K = [[1.88065033e+03, 0.00000000e+00, 3.19678628e+02],

[0.00000000e+00, 1.74384279e+03, 2.40857223e+02],

[0.00000000e+00, 0.00000000e+00, 1.00000000e+00]]

Reconstruction Error: 0.16871827066105438

When 2nd image co-ordinates (last 6 lines of points.txt) are considerd as world co-ordinates, we got camera matrix and re-projection error as:

K = [[1.82200249e+03, 0.00000000e+00, 3.18116241e+02],

[0.00000000e+00, 1.94918334e+03, 2.55714333e+02],

[0.00000000e+00, 0.00000000e+00, 1.00000000e+00]]

Reconstruction Error: 0.14127726078242311

The elements of camera matrix above are in pixel units. We can observe that re-projection error is almost similar in above 3 cases, with slight differences. Focal lengths are different from method1 and method2, because in the method 2 world size will be same as image size. Once we convert it to mm they’ll be same.

**Q4: Were you able to complete this task without the image size? Discuss the presence or absence of this information.**

We have assumed the size to be 640x480 for above results.

The focal length in mm depends on the width and height of the image and the actual object in real world. When we change the size of image, in the method 1 actual size would be constant. In method 2 becuase when we increase image size world size also increases as we are taking one of the image co-ordinates as iworld co-ordinates. Therefore, we can’t observe much difference in fx, fy in method 2, whereas in method 1 you will find huge difference.

We can find significant difference in principal point co-ordinates (cx, cy) in pixel co-ordinates because principal point is the projection of camera center (which would be constant for a particular image) on to the image plane.

Camera matrices for different methods when image size is 1000x1000.

Method 1:

K = [[407.86237031, 0., 510.61893513],

[ 0., 383.81602044, 465.6635847 ],

[ 0., 0., 1. ]]

Reconstruction Error: 0.07146140164329602

Method 2:

Case 1: Image 1 co-ordinates as world co-ordinates

K = [[1.29732025e+03 0.00000000e+00 5.03754208e+02]

[0.00000000e+00 1.34666034e+03 5.04464843e+02]

[0.00000000e+00 0.00000000e+00 1.00000000e+00]]

Reconstruction Error: 0.13409081833527067

Case 2: Image 2 co-ordinates as world co-ordinates

K = [[1.03239538e+03 0.00000000e+00 5.12342224e+02]

[0.00000000e+00 1.01753698e+03 4.93633272e+02]

[0.00000000e+00 0.00000000e+00 1.00000000e+00]]

Reconstruction Error: 0.09239460387063092

We can observe there’s a change in reprojection error too, and we got better results for the case of image size 1000\*1000, which means that camera matrix is estimated successfully.