

I wrote some code to capture frames using depthai api and the OAK D Lite camera
All source code is in the GitHub link: <https://github.com/bdairo/cv-assignment-1>

Question 1

For this question, I used the OAK-D Lite camera to capture 16 checkerboard images then I ran the camera calibration code using the images. The checkerboard consists of 9x7 squares, 25mm each and 8x6 vertices. The calibration process involved detecting specific points on the checkerboard which are known in the real-world space and correlating these points with their corresponding image coordinates. I was able to get the intrinsic parameters of the camera as shown below.

```
Intrinsic Matrix:  
[[442.5930392    0.        294.09394662]  
 [  0.        441.27421154 242.95287843]  
 [  0.          0.          1.          ]]
```

442.5930392 is f_x , the focal length in the x direction.

441.27421154 f_y , the focal length in the y direction.

294.09393662 and 242.95287843 are the O_x and O_y coordinates of the principal point.

Question 2

For this question, I again performed camera calibration using the checkerboard images and got the same results as in Question 1. Using one of the calibration images, I used Open Cv's library for image formation pipeline to compute the rotation along each axis with the results shown below.

```
> python3 Q2.py  
Intrinsic matrix:  
[[442.5930392    0.        294.09394662]  
 [  0.        441.27421154 242.95287843]  
 [  0.          0.          1.          ]]  
Distortion coefficients:  
[[ 4.21607713e-02 -7.20503624e-02  5.16224769e-05 -5.47832421e-03  
  5.03310842e-02]]  
Rotation Angles (degrees):  
X: -0.10, Y: -1.42, Z: -0.25
```

🍏 ~/Py/ComputerVision/HW1 🐱 main !13 ?24

The angles are:

- -0.10 on the X axis indicating that the camera was pointing slightly downwards.
- -1.42 on the Y axis indicating that the camera was turned slightly to the left.
- -0.25 on the Z axis indicates a very minor clockwise rotation from the perspective of looking into the camera.

In addition, the small values for the angles indicate the rotation along each angle was very small.

Question 3

I setup the camera and the coin on a vertical surface. For this setup, I measured the distance between the camera and the coin to be 55.4mm. The coin itself is a quarter with a diameter of 25 mm.

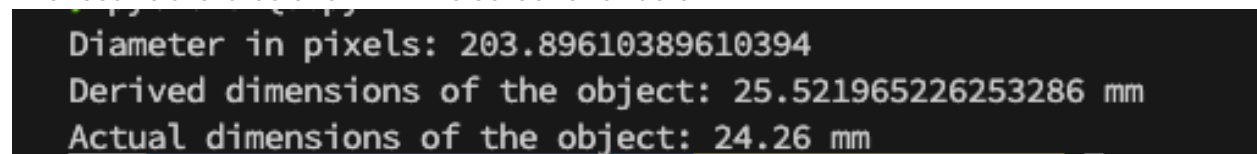
The code was written in python using python matplotlib pltinput() function which is similar to matplotlib ginput() function. This provides a GUI that allows the user to click on two points in the image. After clicking the two farthest points representing the diameter of the coin on the image. The diameter of the coin was calculated using the pinhole camera model where the relationship between the real-world dimensions and their projection in the image depends on the focal length of the camera and the distance of the object to the camera. The focal length was already calculated from Question 1.

After running the code and selecting two points, I got a result of 25.52 mm which is very close to the actual coin diameter of 24.26mm.

Demo to see the application working:

<https://youtu.be/NwJwbZcJvF4?si=RCvMLF1rhH0S5cWq>

The results are also shown in the screenshot below



```
Diameter in pixels: 203.89610389610394
Derived dimensions of the object: 25.521965226253286 mm
Actual dimensions of the object: 24.26 mm
```

Question 4

The application is built using python and flask on the backend and Html/CSS for the front end.

To run the code, navigate to the application directory where app.py is located and run **python3 app.py**

This should open a server where the html file will be served. Navigate to the browser and then upload an image, select points, and input the distance from the camera to the image.

For my setting up my testing, I used an image from the **coin_images** folder and measured the distance from the camera to the image coin as **55.4.mm**

Demo to see the web app working:

https://youtu.be/Mlpvzc2OU_w?si=WE_OFu2oiafUvCXG

The result is shown below:

Dimension Calculation

Upload Image of Object: 1710647661887.png



Distance to Object (mm):

Dimension: 25.53497004944045 mm