CSc 4980/6980: Computer Vision

Assignment 1

Submission in Classroom:

Camera images of paper worksheets will NOT be accepted

Python: submit a jupyter notebook and the .py files associated

MATLAB: submit a MATLAB Live script (.mlx file) and also convert the .mlx file to PDF and append to PDF from Part A.

The MATLAB Live Script document must contain all the solutions, including graphs. The file must be saved as ".mlx" format. See here for live scripts: https://www.mathworks.com/help/matlab/matlab_prog/create-live-scripts.html

Manage all your code in a github repo for each assignment. Provide a link to the repo in the documentation workspace (jupyter notebook or mlx file).

Create a working demonstration of your application and record a screen-recording or a properly captured footage of the working system. Upload the video in the Google classroom submission.

Hardware: Unless otherwise specified, CAMERA refers to the OAK-D Lite camera provided to you.

Software:

MATLAB: Either of the following will work: Use MATLAB R2018b or later version as installed in your machine (installation instructions already provided) **OR** Use MATLAB Online (https://www.mathworks.com/products/matlab-online.html).

For OAK-D you can implement your solutions in either Python or C/C++: https://docs.luxonis.com/en/latest/

PART A: Fundamentals

MATLAB:

1. Go over the camera calibration toolbox demonstration and calibrate the OAK-D camera https://www.mathworks.com/help/vision/ug/single-camera-calibrator-app.html.

Python:

1. Go over the camera calibration toolbox demonstration and calibrate the OAK-D camera

https://docs.opencv.org/4.x/dc/dbb/tutorial_py_calibration.html

PART B: MATLAB/Python Prototyping

2. Write a MATLAB/Python script to find the real world dimensions (e.g. diameter of a ball, side length of a cube) of an object using perspective projection equations. Validate using an experiment where you image an object using your camera from a specific distance (choose any distance but ensure you are able to measure it accurately) between the object and camera.

If using MATLAB: You can use MATLAB's *ginput()* function to record the pixel coordinates of the object's end points on the image.

Example usage:

```
I = imread('rice.png'); % Read the image
imshow(I); % Display the image
  [x y] = ginput(2); % reads two points. x is a 2x1 column vector with
            x coordinates and y is a 2x1 column vector with y
                  coordinates.
```

PART C: Application development

Familiarize with the Depth AI SDK. Run the tutorials/examples:
https://docs.luxonis.com/projects/sdk/en/latest/ (need not submit this)

3. Setup your application to show a RGB stream from the mono camera and a depth map stream from the stereo camera simultaneously. Is it feasible? What is the maximum frame rate and resolution achievable?

4. Run the camera calibration tutorial. Compare the output with answers from Part A calibration results.