CSc 8830: Computer Vision

Homework Assignment 1

Submission in Classroom:

For Part A,

convert your problem solving by hand into a digital format (typed or scanned only. You can use camera scanner apps) and embedded/appended into the final PDF documentation. Camera images of paper worksheets will NOT be accepted

For Part B 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

For Part C, manage all your code in a github repo for each assignment. Provide a link to the repo in the PDF document for Part A. 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, use the OAK-D Lite camera provided to you.
Software: 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/

Preparatory exercises:

(1) Go over the camera calibration toolbox demonstration: https://www.mathworks.com/help/vision/ug/single-camera-calibrator-app.html. Use this exercise to calibrate any camera of your choice.

(2) Execute the Measuring Planar Objects with a Single Camera example: https://www.mathworks.com/help/vision/examples/measuring-planar-objects-with-a-calibrated-camera.html.

PART A: Theory

1. Point the camera to a chessboard pattern or any known set of reference points that lie on the same plane. Capture a series of 10 images by changing the orientation of the camera in each iteration. Select any 1 image, and using the image formation pipeline equation, set up the linear equations in matrix form and solve for intrinsic and extrinsic parameters (extrinsic for that particular orientation). You will need to make measurements of the actual 3D world points, and mark pixel coordinates.

Once you compute the Rotation matrix, you also need to compute the angles of rotation along each axis. Choose your order of rotation based on your experimentation setup.

2. Select any pair of images from the set in problem 1 above. Compute the homography between those two images.

PART B: MATLAB Prototyping

3. Write a MATLAB 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.

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

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

5. 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?

6. Run the camera calibration tutorial. Compare the output with answers from Part A and Matlab calibration exercise.