# COMP 4102A: Assignment 2 - Part 1

Posted: February 23,2020 Due by 23:59 Sunday, March 15

## 1 (20 points) Theory questions

1. (5 mark) Does a pin-hole camera (no lens) have an infinite depth of field or not? Explain the answer in a short English sentence.

2. (5 mark) A given fixed point $P = (X0, Y0, Z0)$ projects to the image plane at a 2d point $p = (x, y)$. There is a 3d line from the origin through the 3d point P. Prove that every point on that line projects to the same 2d point p. The two projection equations are $x = fX/Z$ and $y = fY/Z$. Hint: First write down the parametric equations of all possible points X, Y, Z on the line from the origin through the 3d point P. These are three parametric equations, one for each of X, Y and Z. Now use these three parametric equations along with the projection equations to prove that every point on this line projects to the same 2d point p.

3. (10 mark) A pinhole camera has focal length $f = 500$ , pixel sizes $S_x = S_y = 1$ , and its principal point is at $(o_x, o_y) = (320, 240)$. The world coordinate frame and the camera coordinate frame can be related by $X_c = RX_w + T$ , where

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, T = \begin{bmatrix} -70 \\ -95 \\ -120 \end{bmatrix}$$

(a) Write out the $3 \times 4$ projection matrix that projects a point in the world coordinate frame onto the image plane in pixel coordinate.
(b) What are the pixel coordinates of the world point $X_w = [150 \quad 200 \quad 400]^T$

## 2 (15 mark) Projection 3d to 2d

Using the same $R, T, f, s_x, s_y$, and $X_w$ as given in Question 3 write an OpenCV program that projects this single 3d point into 2d pixels using the given camera parameters. The routine projectpoints() that you must use is described in https://docs.opencv.org/3.4/d9/d0c/group__calib3d.html.Remember to also include include "opencv2/calib3d/calib3d.hpp" in your program to be able to use projectpoints(). I first set up K and R matrix, and T vector, along with a zero distortion vector; all as mats of type float (using the C++ interface for OpenCV). It also seems to work if they are allocated as doubles. Look at the Mat tutorial in OpenCV and remember to create all vectors as the appropriate sized column array. I declare a single 3d point which is to be projected to 2d as a vector<points3f> as described in the documentation for projectpoints. Since the rotation matrix is the identity matrix you do not need to convert this matrix to the rotation vector format by calling the routine Rodrigue. Instead you can simply pass a zero rotation vector to the routine projectpoints. In Python you access OpenCV by using import cv2, not import cv (which does not work in Python). Also, in python the input vectors should be row

vectors and not column vectors, which is opposite from C++. Also the arrays should be explicitly declared using numpy as floats, or initialized to floating values (same result). Your final program should print or display the value of $X_w$ along with the projected x and y pixel positions. The final projected pixel values should be very close to what you calculated by hand in Question 3. Include the source code of your program and some proof that the program produces the correct results (either copy the printed output as a picture using screen capture or print the results to a file). A simple tutorial on writing text to files is in http://www.cplusplus.com/doc/tutorial/files/. Please do not include the program executable. Do not create a complete projection matrix by hand, please use the routine projectpoints to compute the 2d projection of the given 3d point.

## 3 (15 points) Harris Corner detector

The goal of this assignment is to use the routines in OpenCV to demonstrate the operations of the Harris Corner detector. The program should do the following:

1. Open the image box-in-scene.jpg and show the image in a window.
2. Compute the minimum eigenvalue of that image. Hint: The easiest way is to use the routine cvCornerMinEigenVal.
3. Threshold the minimum eigenvalue, and draw the pixels that pass this threshold test in white, and the rest of the pixels in black. The actual threshold should be set by a slider which is on top of the window. (Try a threshold around the value of 0.01 for the minimum eigenvalue which would gives good results as the middle value of the slider.)
4. Take the pixels that pass this threshold test and use a non-maxima suppression algorithm to thin out the potential corners. Draw the corners that pass the non-maxima suppression test in another window. These are the final corners. You can show these corners with small circles or cross.

As you vary the minimum eigenvalue threshold you will see that the number of detected corners changes. The code in https://docs.opencv.org/3.4/da/d6a/tutorial_trackbar.html is an example of how to use a slider in HighGUI.

## Submission

Please write a document (pdf or doc) with your solutions on theory questions. Include your codes for Projection 3d to 2d and Harris corner detector separately in two folders. Submit a zipped folder contains: theory questions solutions, codes for Projection 3d to 2d and Harris corner detector. For Harris corner detector, include snapshot of the slider, images before and after applying non-maxima suppression and final result with detected corners. Please submit through cuLearn. You are expected to work on the assignment **individually**.