ENPM673 – Perception for Autonomous Robots

Homework 1

Srikumar Muralidharan, Samer Charifa

Due date: 15th February 2021, 11:59PM

Submission guidelines:

- This homework is to be done and submitted individually.
- Your submission on ELMS/Canvas must be a zip file, following the naming convention **YourDirectoryID_hw1.zip**. If you email ID is abc@umd.edu or abc@terpmail.umd.edu, then your Directory ID is **abc**. Remember, this is your directory ID and NOT your UID.
- Please provide detailed instructions on how to run your code in README.md file.
- For each section of the homework, explain briefly what you did, and describe any interesting problems you encountered and/or solutions you implemented.
- For Problem 3, you are not allowed to use built in functions for computing SVD and Pseudoinverse directly. However, use of low-level inbuilt functions like function for computing eigen vectors and transpose are allowed.

Problem 1: **[20 Points]**

Assume that you have a camera with a resolution of 9MP where the camera sensor is square shaped with a width of 14mm. It is also given that the focal length of the camera is 15mm.

1. Compute the Field of View of the camera in the horizontal and vertical direction.
2. Assuming you are detecting a square shaped object with width 5cm, placed at a distance of 20 meters from the camera, compute the minimum number of pixels that the object will occupy in the image.

Problem 2: **[60 Points]**

A ball is thrown against a white background and a camera sensor is used to track its trajectory. We have a near perfect sensor tracking the ball in video1 and the second sensor is faulty and tracks the ball as shown in video2. Clearly, there is no noise added to the first video whereas there is significant noise in video 2. Assuming that the trajectory of the ball follows the equation of a parabola:

- Use Standard Least Squares, TLS and RANSAC methods to fit curves to the given videos in each case. You have to plot the data and your best fit curve for each case. Submit your code along with the instructions to run it. (Hint: Read the video frame by frame using OpenCV's inbuilt function. For each frame, filter the red channel for the ball and

detect the topmost and bottom most colored pixel and store it as X and Y coordinates. Use this information to plot curves.) **[40 points]**

- Briefly explain all the steps of your solution and discuss which would be a better choice of outlier rejection technique for each case. **[20 Points]**

Problem 3: **[20 Points]**

The concept of homography in Computer Vision is used to understand, explain and study visual perspective, and, specifically, the difference in appearance of two plane objects viewed from different points of view. This concept will be taught in more detail in the coming lectures. For now, you just need to know that given 4 corresponding points on the two different planes, the homography between them is computed using the following system of equations Ax = 0, where:

$$A = \begin{bmatrix} -x1 & -y1 & -1 & 0 & 0 & 0 & x1 * xp1 & y1 * xp1 & xp1 \\ 0 & 0 & 0 & -x1 & -y1 & -1 & x1 * yp1 & y1 * yp1 & yp1 \\ -x2 & -y2 & -1 & 0 & 0 & 0 & x2 * xp2 & y2 * xp2 & xp2 \\ 0 & 0 & 0 & -x2 & -y2 & -1 & x2 * yp2 & y2 * yp2 & yp2 \\ -x3 & -y3 & -1 & 0 & 0 & 0 & x3 * xp3 & y3 * xp3 & xp3 \\ 0 & 0 & 0 & -x3 & -y3 & -1 & x3 * yp3 & y3 * yp3 & yp3 \\ -x4 & -y4 & -1 & 0 & 0 & 0 & x4 * xp4 & y4 * xp4 & xp4 \\ 0 & 0 & 0 & -x4 & -y4 & -1 & x4 * yp4 & y4 * yp4 & yp4 \end{bmatrix}, x = \begin{bmatrix} H_{11} \\ H_{12} \\ H_{13} \\ H_{21} \\ H_{22} \\ H_{23} \\ H_{31} \\ H_{32} \\ H_{33} \end{bmatrix}$$

For the given point correspondences,

	x	y	xp	yp
1	5	5	100	100
2	150	5	200	80
3	150	150	220	80
4	5	150	100	200

Find the homography matrix:

$$H = \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{bmatrix}$$

- Show mathematically how you will compute the SVD for the matrix A.
- Write python code to compute the SVD.