CS4277/CS5477 ASSIGNMENT 3: AFFINE 3D MEASUREMENT FROM VANISHING LINE AND POINT

1. OVERVIEW

In the third assignment, you will be finding the height of a target object given the height of a query object. We will be:

1. Finding vanishing points within an image using RANSAC.
2. Obtain affine 3D measurements from a vanishing line and vanishing point.

References: Lecture 6.

Honour Code. This coding assignment constitutes **10%** of your final grade in CS4277/CS5477. Note that plagiarism will not be condoned! You may discuss with your classmates and check the internet for references, but you MUST NOT submit code/report that is copied directly from other sources!

2. SUBMISSION INSTRUCTIONS

Items to be submitted:

- **Source code (main.py).** This is where you fill in all your code.
- **Report (report.pdf).** This should describe your implementation and be no more than one page.

Please clearly indicate your name and student number (the one that looks like A1234567X) in the report as well as the top of your source code. Zip the two files together and name it in the following format: **A1234567X_lab3.zip** (replace with your student number). Opening the zip file should show similar to lab1.

Submit your assignment by **2 March 2021, 2359HRS** to LumiNUS. 25% of the total score will be deducted for each day of late submission.

3. GETTING STARTED

This assignment as well as the subsequent ones require Python 3.7, or later. You need certain python packages, which can be installed using the following command:

```
pip install -r requirements.txt
```

If you have any issues with the installation, please post them in the forum, so that other students or the instructors can help accordingly.

4. Problem Statement

You work as a computer vision engineer at a house mover company i.e., the people that help you to move your furniture from your previous home to your new home. Your boss has several trucks to help with the move but he is concerned that the ceiling of the carpark is too low for his trucks. Knowing that you have taken CS4277/CS5477, you boss asks if you are able to calculate the height of the carpark (in red) from a google maps image of the clients' new home in Figure 1.

Figure 1: Query and target heights

Being the ingenious engineer, you said yes and decided to use the height of the red minivan to determine the height of carpark. You found that the height of similar mini-vans i.e., 2008 Dodge Grand Caravan is 1.75 metres tall[1]. Your algorithm has two steps: (1) First, you will find three vanishing points in the image (two horizontal vanishing points and one vertical vanishing point); (2) Then, using the horizontal vanishing points, you create a vanishing line and perform 3D affine measurements from the horizontal vanishing line and the vertical vanishing point.

5. Finding Vanishing Points

We will use RANSAC to compute the vanishing points. First, we detect lines in the image using the Canny Edge Detector and the Hough Transform from the OpenCV library to detect the best lines i.e. Figure 2.

Figure 2: detected lines.

Once the lines have been detected, we separate the vertical lines from the non-vertical lines i.e., Figure 3.

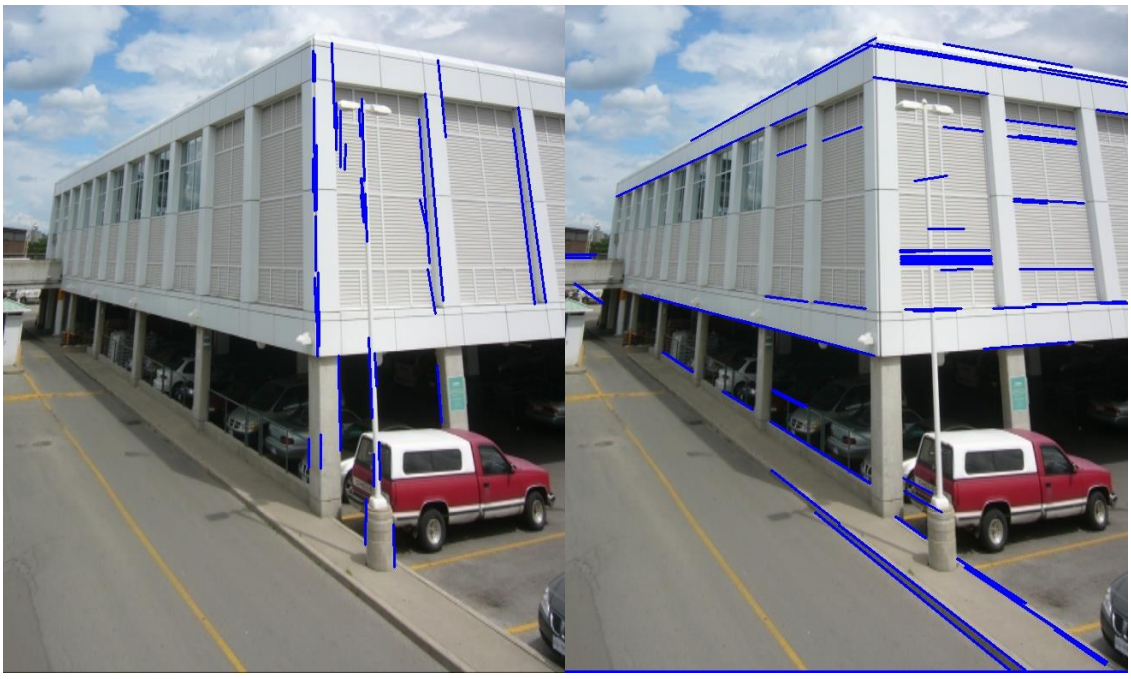

Figure 3: vertical lines (left); non-vertical lines (right).

Then, we compute two horizontal vanishing points from the non-vertical lines and a single vertical vanishing point from the vertical lines. To find n number of vanishing points e.g. $n = 2$ for the number of horizontal vanishing points, we will use the following RANSAC algorithm.

1. From L lines, we compute the pairwise intersections between each pair of lines.

2. Then, we compute the support matrix i.e., M[i,j]=1 implies that the intersection indexed at i is supported by the line indexed at [j]. We define that the line supports the intersection if the shortest distance between the line and the intersection (point) is within a distance threshold.
3. For n loops:
    a. We select the intersection with the largest support set as a vanishing point.
    b. We remove the lines in the support set of the select intersection (point) from the support sets of other intersections.

The RANSAC algorithm will be ran separately from the vertical lines where $n = 1$ and the non-vertical lines where $n = 2$, i.e., Figure 4.

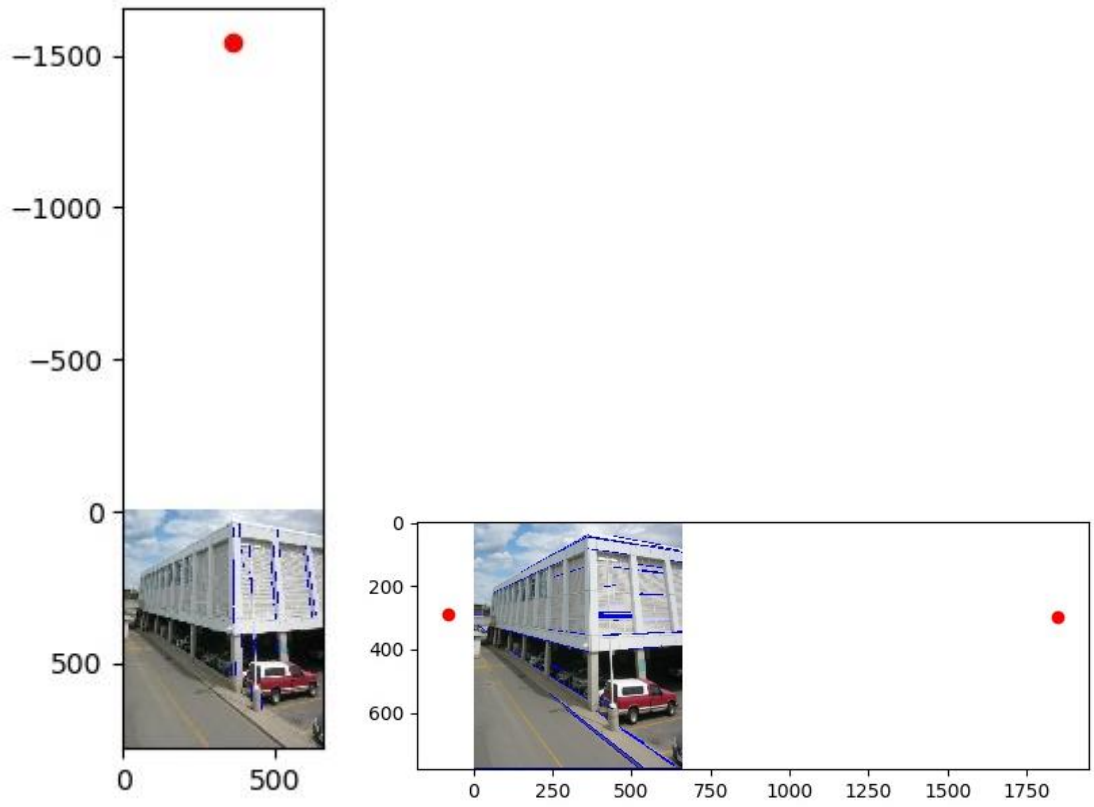

Figure 4: vertical vanishing point (left); horizontal vanishing points (right).

6. 3D Affine Measurements

From our horizontal vanishing points, we compute the horizontal vanishing line. Using the horizontal vanishing line, vertical vanishing point, minivan height, and image coordinates of the pillar and minivan, we will obtain the height of the pillar i.e., carpark. From lecture 6, we perform the following steps:

1. We compute the vanishing point u from the bottom coordinates of the pillar and the minivan.

2. Then, we compute the transferred point of the top coordinate of the minivan onto the line passing through the coordinates of the pillar and the vertical vanishing point.
3. Compute the distance of the vertical vanishing point, top pillar image coordinate and transferred point from the bottom pillar coordinate.
4. Compute the distance ratio between the height of the minivan and the height of the pillar.
5. Compute the height of the pillar as the product of the distance ratio and the height of the minivan.

7. ASSIGNMENT TASK

Your task in this assignment is to implement the following functions for the algorithm. We will provide the skeleton file with missing code. You are to implement the following:

- `detect_lines [1]`: detect the lines in the image using the Canny Edge Detector and the Hough Line Transform.
- `get_pairwise_intersections [1]`: Compute the intersections between pairs of detected lines, removing lines that do not intersect i.e., intersection points at infinity.
- `get_support_mtx [2]`: From the pairwise intersections and lines, compute the support matrix between the intersections and the lines if the distance between the intersection and line is within the distance threshold.
- `get_vanishing_pts [2]`: Perform the RANSAC algorithm loop to find vanishing points.
- `get_vanishing_line [1]`: Compute the vanishing line from the pair of horizontal vanishing lines.
- `get_target_height [3]`: Performs 3D affine measurements algorithm from vanishing line and points in Lecture 6 to obtain the height of the pillar (carpark) from the height of the red minivan.

Write your code with the block (“”” YOUR CODE HERE “””) and (“”” END YOUR CODE HERE “”””) for each function. Students that do not follow submission instructions will incur a 1-point penalty. Students are strongly encouraged to seek help via the forums. We also provide the outputs from our implementation in `ta-results/`. You are strongly encouraged to start your implementation early.