

3630 Project 2: Multi-Modal Sensing

Released Feb 6, 2023

Due Feb 19, 2023

Overview

The setting for this project is that an [e-puck robot](#) is navigating a maze to find a traffic sign. You are provided code that will determine whether the traffic sign is in the camera feed of the robot. Your task will be to determine:

1. The depth (not the distance) and angle to the sign using the 2-cameras method
2. The distance and angle to the sign using camera+lidar method

There are two identical cameras attached to the robot, horizontally displaced from each other. First, you will determine the distance using the stereo camera feed. You will use the two images captured simultaneously from these two cameras to determine the distance and heading from the traffic sign using a triangulation method. The L7_Camera.ppt lecture notes (found in Canvas) should be helpful for accomplishing this portion of the assignment.

In the second part, you will leverage the lidar on the robot to determine the distance and heading, using both the point cloud of the lidar, and one image captured by the robot. The lidar data is composed of an array of 360 distance values around the robot, one for each degree for 0 to 359 deg.

Webots Simulator

We will be employing the [Webots](#) simulator in this project (and in some of the future projects in this course). You will need to install it using the following [installation procedure](#) (this includes documentation for Linux, Windows, and Mac).

You will be provided with a series of world files and a controller for the e-puck robot. The assignment will then require finding the distance/depth and heading of a traffic sign, relative to the robot, in each world file. The code structure and respective functions is described in the next section.

To test out your scripts in the Webots simulator, open a world file (named either as proj2_world_vision_lidar_#.wbt or as Vision_only_world#.wbt), and run the simulation. The first few (typically 3) iterations running the controller might have blank images, however it should then allow you to see the required distance/ depth and angle values print on the Webots console.

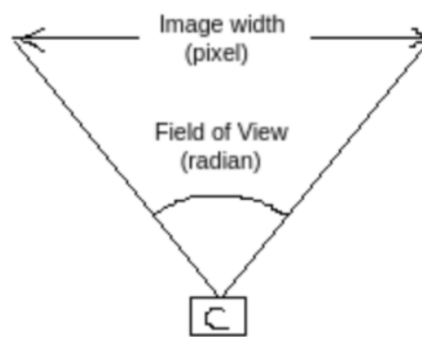
Requirements Installation

You can use the requirements.txt file to install all necessary packages by running `pip install -r requirements.txt`

Assignment

Preliminaries

The field of view is the range of the world (in this project, defined in radians) in the horizontal plane that is visible.



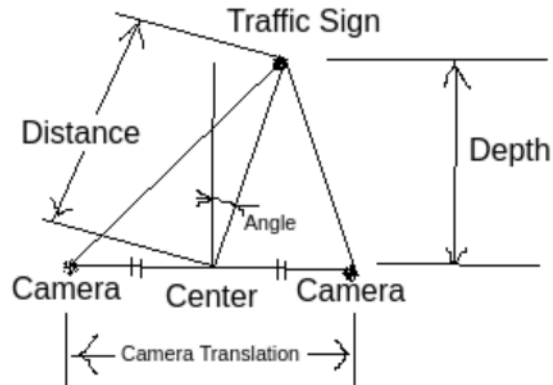
We provide the following formula that can be used to include the field of view in the calculation of focal length for both parts 1 and 2.

$$\text{focal_length} = \text{image_width} / (2 * \text{math.tan}(\text{fov} / 2))$$

Part 1: Determine depth and angle using stereo vision

To complete part 1, you will need to fill out the `vision_only_depth_calculation(image1, image2, fov, camera_translation)` function in `proj2_vision_only_controller/vision_only_calculation.py`

The `vision_only_depth_calculation(image1, image2, fov, camera_translation)` function takes in two images, camera field of view, camera translation (distance between the 2 cameras) and returns the depth (perpendicular distance from the traffic sign to the line joining the cameras) and angle from the robot to the traffic sign in the images.

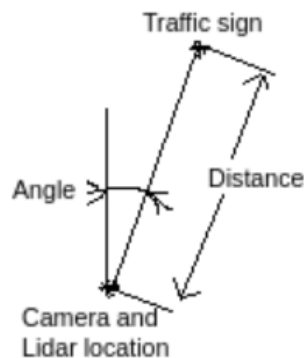


Hint: try using trigonometry to calculate the angle, using the parameters mentioned in the diagram above.

Part 2: Determine distance and angle using vision and lidar

To complete part 2, you will need to fill out the `vision_lidar_distance_calculation(image, lidar_range_array, fov)` function in `proj2_vision_lidar_controller/vision_lidar_calculation.py`

The `vision_lidar_distance_calculation(image, lidar_range_array, fov)` function takes in an image and the point cloud returned from a lidar, and returns the distance and angle from the robot to the traffic sign in the images.



Functions Provided

To complete the above two parts, you are provided with the `contour.py` file, which contains a function `box_measure(image)` that calculates the pixel values of the centroid of the identified traffic sign in the image. This file (identical copies) can be found in both the `proj2_vision_lidar_controller` and the `proj2_vision_only_controller` folders.

Submission Instructions

The files you will submit to Gradescope are `vision_lidar_calculation.py` and `vision_only_calculation.py`

To test your solution locally, you can employ the provided `local_tests.py` files.

You are allowed to employ the following two libraries. Note that you may not use any libraries outside this list.

- NumPy
- OpenCV

Rubric for Vision Component

Your solution for part 1 (stereo vision) will be evaluated in 5 world files. Each world will be worth 10 points for a total of 50 points possible.

Depth Evaluation

For each world, you will receive a number of points based upon how close your detected depth is from the true depth. The difference between your depth and the true depth will be evaluated as a percentage of the true depth.

For example, if the true depth is 10 cm and you say that the depth is 9 cm, then the difference is 1 cm. This is 10% of the true 10 cm, so the grade would be 8 out of 10 for that world file.

Percentage difference in distance	Point Count
<= 5% of true distance	5 points (full credit)
<= 10% of true distance	4
<= 15% of true distance	3
<= 20% of true distance	2
<= 25% of true distance	1
> 25% of true distance	0

Angle (Heading) Evaluation

For each world, you will receive a number of points based upon how close your detected angle is from the true angle. The difference between your angle and the true angle will be evaluated as difference in degrees.

Degrees difference in heading	Point Count
<= 5 degrees	5 points (full credit)

<= 10 degrees	3.75
<= 15 degrees	2.5
<= 20 degrees	1.25
> 20 degrees	0

Rubric for Vision + Lidar Component

Similarly, your solution for part 2 (vision + lidar) will be evaluated in 5 world files. Each world will be worth 10 points for a total of 50 points possible for part 2.

Distance Evaluation

For each world, you will receive a number of points based upon how close your detected distance is from the true distance. The difference between your distance and the true distance will be evaluated as a percentage of the true distance.

Percentage difference in distance	Point Count
<= 5% of true distance	2 points (full credit)
<= 10% of true distance	1.5
<= 15% of true distance	1
<= 20% of true distance	0.5
> 20% of true distance	0

Angle (Heading) Evaluation

For each world, you will receive a number of points based upon how close your detected angle is from the true angle. The difference between your angle and the true angle will be evaluated as difference in degrees.

Degrees difference in heading	Point Count
<= 5 degrees	8 points (full credit)
<= 10 degrees	6
<= 15 degrees	4
<= 20 degrees	2
> 20 degrees	0