Introduction to Robotics Institut für Informatik, Freie Universität Berlin
WiSe 21/21 Prof. Daniel Goehring, Stephan Sundermann, Khaled Alomari

Exercise 4 November 8, 2021

Submission online until **Tuesday, 23.11.2021, 11:55 a.m**

Objective: In this exercise you will write a program to read white points in the floor with the RGBD camera of the car and use those points to compute the location and orientation of the camera in space. In order to achieve this objective, functions from the OpenCV library must be used.

Assignment 4-1: Field Preparation

Play the provided bagfile in a loop:

```
rosbag play -l calibration.bag
```

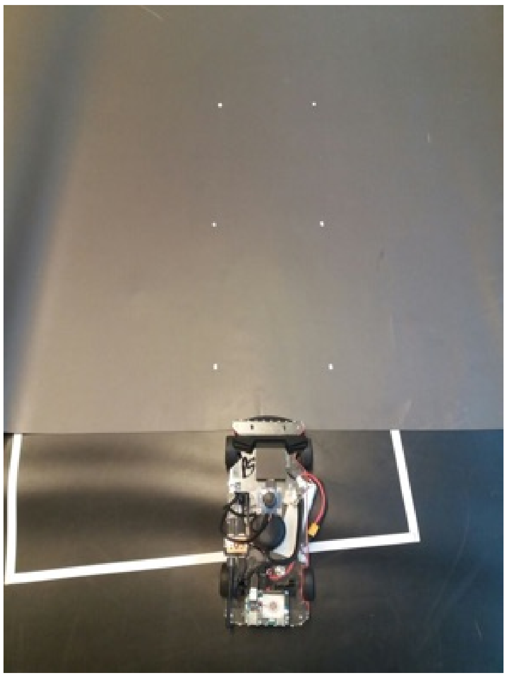


Figure 1: Prepared field

Assignment 4-2: Camera parameters (2 Points)

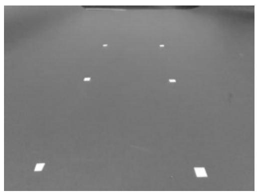
Take a look at the definition of the `sensor_msgs/CameraInfo` message type:

`http://docs.ros.org/noetic/api/sensor_msgs/html/msg/CameraInfo.html`

Extract the intrinsic parameters f_x, f_y, c_x, c_y and distortion coefficients k_1, k_2, t_1, t_2, k_3 from the `/sensors/camera/infra1/camera_info` topic.

Assignment 4-3: Binary Image (4 Points)

Use the `threshold` function from OpenCV to turn your gray image into binary image. Adjust the intervals until you find a good threshold where the white marks are separated nicely. If white pixels other than the calibration points remain, you can mask them by drawing black rectangles. Publish your image. (see Fig. 2b)


(a) Greyscale image

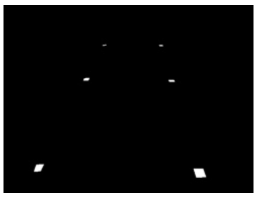

(b) Binary image

Figure 2: Images

Assignment 4-4: Find white pixels (4 Points)

For simplicity you can manually divide the thresholded image into 6 regions so that every region has one marker. For every region extract all the white pixels and calculate the average position to get the center of the marking. It is also allowed to use a clustering algorithm from a library for this task. Remember that the coordinates of the image are as follows:

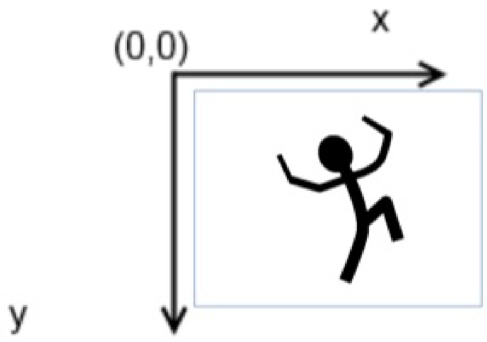


Figure 3: OpenCV image coordinates

Assignment 4-5: Compute the extrinsic parameters (6 Points)

Here you will have to use the `solvePnP` function of OpenCV to calculate the extrinsic camera parameters using the points obtained in the last part. Take a look at the documentation at:

`https://docs.opencv.org/2.4/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html`

Define a 3x3 numpy array for the intrinsic parameters and use the values from task 1. Create a 5x1 numpy array for the distortion parameters retrieved from task 1.

Use the marker centers from the previous task and map them to the real world coordinates. The real world coordinates for the images are as follows, with the world origin being in the car's base_link frame which is the center of the rear axle projected onto the ground:

bottom left	$(0.5m, 0.2m)$
bottom right	$(0.5m, -0.2m)$
middle left	$(0.8m, 0.2m)$
middle right	$(0.8m, -0.2)$
top left	$(1.1m, 0.2m)$
top right	$(1.1m, -0.2m)$

Note, the order of the image points and the real world coordinates passed to `solvePnP` has to match.

Print your results to the terminal.

Assignment 4-6: Finding the camera pose (4 Points)

From the previous task you got the `rvec` and `tvec` parameters. Use the OpenCV function `Rodrigues` to convert `rvec` into a rotation matrix.

- What is the homogeneous 4x4 transformation matrix?
- In which frame is the resulting transformation?
- What is its inverse?
- Which value in which matrix represents the camera height above the ground plane?