Introduction to Robotics WS18/19 – Version 1.01

9. Assignment: Map and GPS

Please summarize your results (images and descriptions) in a pdf-document and name it, e.g., "RO-09-<surnames of the students - group name>.pdf".
There should not be any source code in the pdf document.
Source code will have to be submitted to your repository, provide the link to the repo in the Pdf.
Only one member of the group must submit the document.
Do not copy solutions to other groups.

By the end of this class you will need 60% of points.

1. Map description (2 Points)

Given is the following map of the track, seen from the ceiling camera. The origin is in the upper left of the image, which corresponds to the window side in the lab.

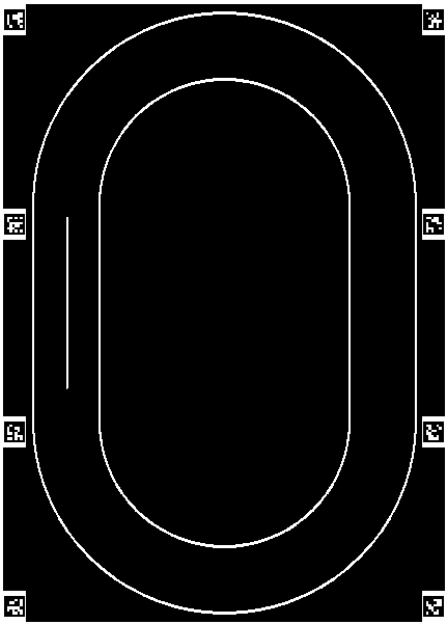
The x-axis points downward (w.r.t. the image shown), the y-axis points to the right. 1 pixel in the image corresponds to 0.01 m in reality.

Your task is to describe the inner white track (inner oval) geometrically. Therefore you can divide the map into 4 parts – 2 half circles and two straight lines.

Hint: For the description of the half circles it is useful to find out, where there center points and there radii are.

You can use the open source software gimp to find the pixel coordinates of a certain pixel.

Provide the equations which you used in your Pdf.



2. Closest point on trajectory (3 Points)

Implement a method (a function) which uses your formulas from task 1 and returns for every possible position on the 2D-plane the closest point on the oval – the oval which you just described in task 1. There should be a unique solution for almost all points – except for the radii of the circles, here you would have infinitely many solutions, i.e., every point on the circle. Make an assumption here for which point to return.

Hint: It can be useful to check at first on which of the four parts of your oval the closest point is, by using the angle of the given point w.r.t. the center points of the circles. Or you can divide the plane into useful intervals w.r.t. x- and y-coordinates. The rest should be as easy as pie.

Provide the code in your repository.

What would be the closest points on the oval for the following 4 points (coordinates in meters)?

1. (0,0)
2. (2, 4)
3. (1, 3)
4. your own choice

Describe your approach in a few sentences in the Pdf, write the results in your Pdf.

3. Ceiling camera GPS (5 Points):

Now read the position of your model car using the ceiling camera server.

Let your car drive one lap on the inner oval – the one for which you have the closest point method in task 2. At each time step read the position of your vehicle from the server and calculate the distance to the closest point on the track.

Once you finished the whole lap, calculate the average absolute distance and the average squared distance to the oval – using your closest point formula from task 2.

Provide the code in your repository. Take a video of how the car went around the oval and submit it to the KVV (5 MB max and mp4).

Plot the trajectory of your vehicle as a 2D-plot (x- and y-coordinates in one plot) and include the plot in your Pdf. In addition, write the average absolute and the average squared distance after driving around the oval in your Pdf.

Hint: If for some reason your model car cannot go around the track by following the lane, you can try to move it carefully by hand around the track (use a platform for the car, do not just push the car with the wheels on the ground as this could damage the motor and/or the gear).

It might be useful to record a bagfile of the ceiling camera localization and having your car performing one lap. Then you can use that bagfile for the rest. How to read the camera localization is described on the next page.

How to use the ceiling camera localization

To start on the Lab-Computer (fisker):

```
export ROS_HOSTNAME=192.168.43.31
export ROS_MASTER_URI=http://192.168.43.31:11311/
roslaunch camera_localization RoboticsLabLocalization.launch
```

You can check in rviz, whether the camera positions are correct (use TF display plugin). The tutors can help you with that. Sometimes the webcams hang up (after 1-2 days). Then restart the cameras by pulling the plugs.

On the vehicle:

There is a launchfile on the vehicle which you have to start.

On the car they have to check that the IP of the host match with the IP on
the localization launch file:

$rosed manual_control Localization.launch

Change the IP of the host:
<rosparam param="sync_hosts">['192.168.43.31']</rosparam>

After running the launch file the topics apear in the topic lists:

rosed manual_control Localization.launch .

On fisker, under camera_localization/rviz is a nice config file for rviz, whichs visualizes the frames.

Then you will receive all topics from the lab computer (fisker) on the model cars.

On the vehicless there is a small number next to the barcode with the id of the Aruco-code. The mapping of IP-address to Acuro-code id is as follows:

IP -> TOPIC
125 -> 5
129 -> 9
122 -> 12
127 -> 7
128 -> 8
121 -> 11
124 -> 4
126 -> 6
120 -> 10
123 -> 3

The topic you want to read on the model car is /localization/odom/{ID} . It should arrive with 30 Hz normally. If the Wifi is slow or the localization cannot perceive the marker, the rate can drop. The topics are of type nav_msgs::Odometry . Position und velocity are set.

In case one of you shut down the fisker-computer accidentally or changed the session, or if the cameras crashes (LEDs on the cameras are off) and you need to restart you will have to compile your own catkin_ws with the following two packages:

https://github.com/AutoModelCar/model_car/tree/version-4.0/catkin_ws/src/map_publisher

https://github.com/stsundermann/camera_localization