Introduction to Robotics WS18/19 – Version 1.00

7. Assignment: Line detection

1. Control a car on a straight lane (10 Points):

Prepare the field if it has not been prepared already, put a white line on the field 5 meters long (or the whole width of the field). Use the line tape with Velcro / Klettband.

a. Recognize the white line or selected parts of the line in the camera image using RANSAC or Hough transform or any other algorithm which you prefer.

Now control the steering of a model car in order to stay on a white line for 4 meters – using the recognized line in the camera as feedback. Do not use any other sensory data, only the camera data. What information from the camera data you use is up to you, but here are some suggestions.

Apply a fixed velocity of roughly 0.1 meters per second or less. To keep things simple, do not control the velocity of your vehicle using a controller, just set a fixed velocity.

Now, how to find out, where to steer:

For example, you can calculate the position of the white line in your image on a certain image row (intersection of the white line and a certain image row) and use the distance of the intersection point to the middle column as the error for your controller.

Or you can take the orientation of the line and its distance to the image center into account. Or do something more fancy. Whatever you do, you will end up with some metric for your error which goes into your controller.

Start with a simple P-controller, if necessary you can later use a PD-controller.

Important: Make sure that the car is not steering all the time or unnecessarily, which can cause damage the steer motor. To avoid permanent oscillations, caused by a changing error over time while the car is not even moving, you can calculate the average, e.g., the average error over time (e.g. 10 time steps). Also reducing the K_P-values (and K_D) will help to keep steering commands small. Think about your initial K_P value: An error of 10 pixels of your line position shall result in what steer command?

Before you start driving just shift the car w.r.t. the white line and see how the steer angle reacts. What you want to see are smooth (not abrupt) steer commands in the direction of the line and also no high frequency oscillations.

Also make sure that your u_t (the result of your steer controller) has to be mapped to a steer command – using the function which you created in assignment 6 or make up a new one. I.e., if your controller outputs 0, the car shall probably steer straight and not fully right.

For your submbission you will perform 2 experiments. Start the car:

1. on the line and let it drive for 4 meters.
2. 0.2 meters to the right of but still facing a direction which is parallel to your line, see if it finds back to the line and how its position converges w.r.t. the line.

Submit your code to the repository, plot your steer commands over time for both experiments (i.e. with ros rqt) and put them into your Pdf and take a video of your two experiments (< 5 MB each, format mp4).

Optional – Or how weights are set up in an easier way: If you want to modify the parameters of your PD-controller with a litte gui and without having to restart your python node each time after updating the weights, it is rewarding to take a look as this tutorial:

http://wiki.ros.org/dynamic_reconfigure/Tutorials

Especially the first two links (mainly about defining the parameters which you want to setup and then changing them):

http://wiki.ros.org/dynamic_reconfigure/Tutorials/HowToWriteYourFirstCfgFile

http://wiki.ros.org/dynamic_reconfigure/Tutorials/SettingUpDynamicReconfigureForANode%28python%29

Once you are done: In the gui it is sometimes helpful to press enter to have a weight change take effect.