Robotiks WS17/17

Assignment 5

Name	MatrNr	Mail
Rémi Toudic	4318284	remitoudic@gmail.com
Sven Heinrichsen	4780388	s.heinrichsen@fu-berlin.de
Alexander Hinze-Huettl	4578322	hinze.alex@gmail.com

Repo: https://github.com/al-eax/robotik_ws1718

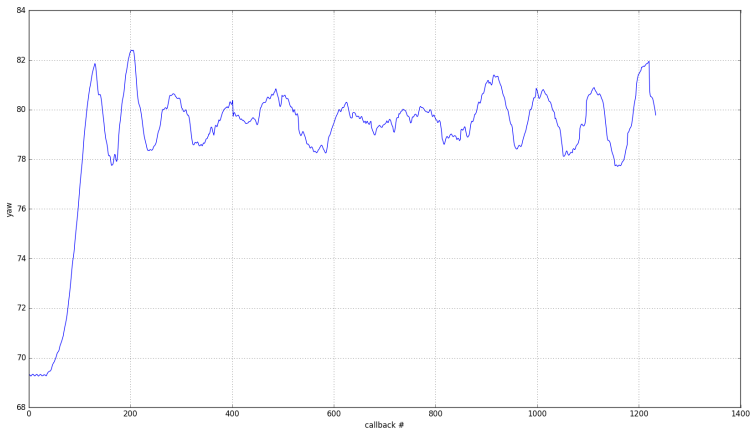
Control a car on a straight lane using a heading sensor

We subscribed to the topic `/model_car/yaw`
We calculated `u` wit the given formular:

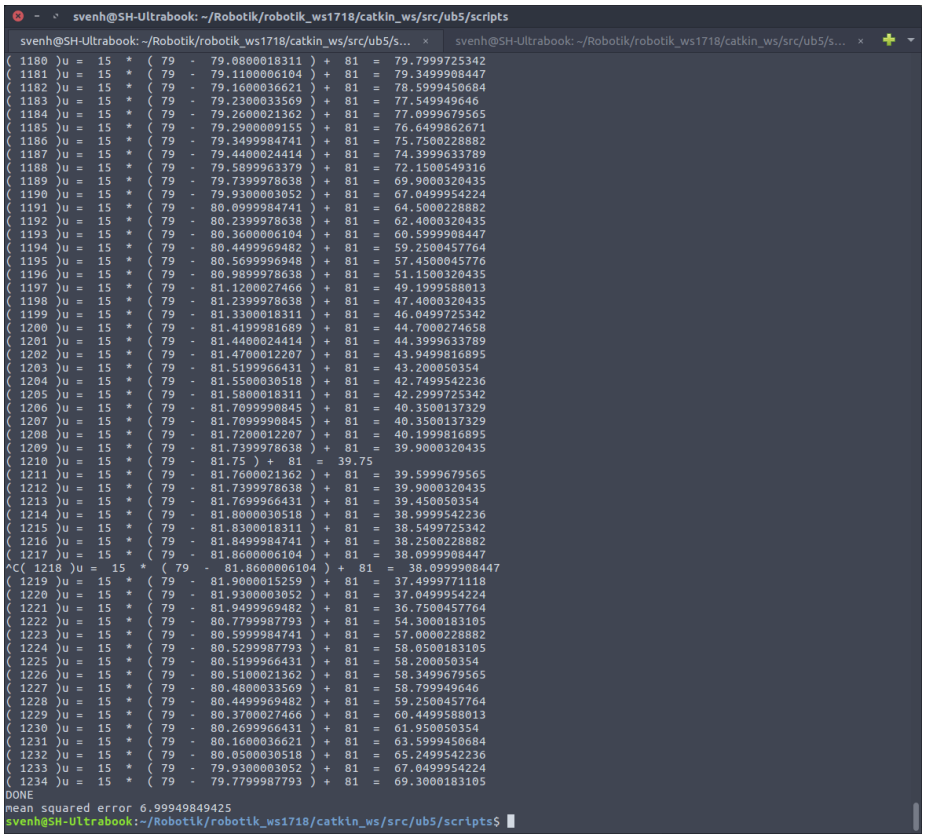
```
u = KP *(desired_yaw - current_yaw) + CALIBRATED_ZERO_ANGLE
```

We switched between a real car and the gazebo simulator. For the real car we used higher weight values because of the limited space in the lab.

This is our plot of the current yaw against the callbacks for the real car with `KP = 15`:



After 1234 callbacks we had a mean suared error of 7. The car starts with a yaw of 70° and oscillates around 79°:

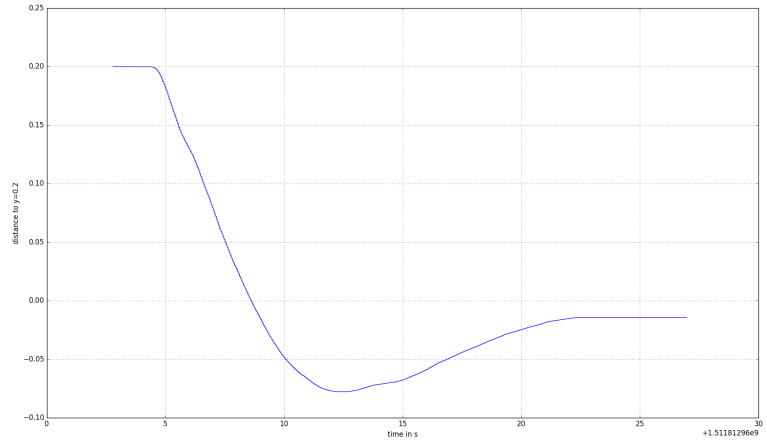


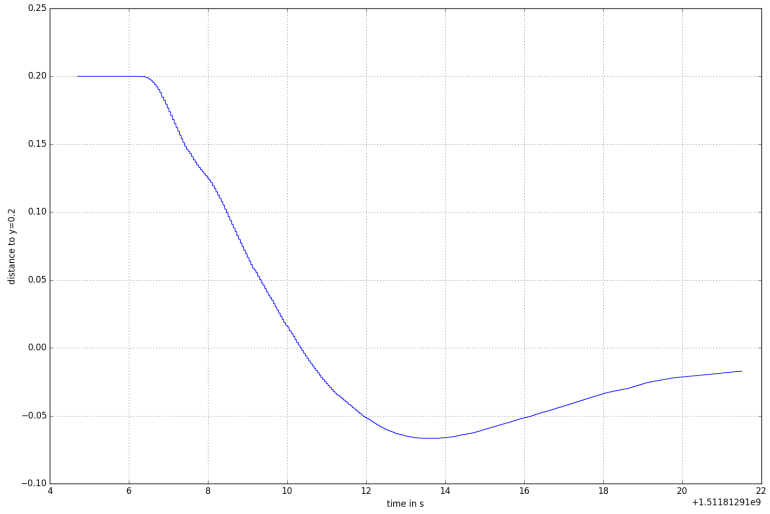
Control a car on a trajectory via odometry

For this task, we subscribed the `/odom` topic. We also recorded the timestamp in `t` and the difference between current y and desired y in `odom_arr` to calculate the derivative:

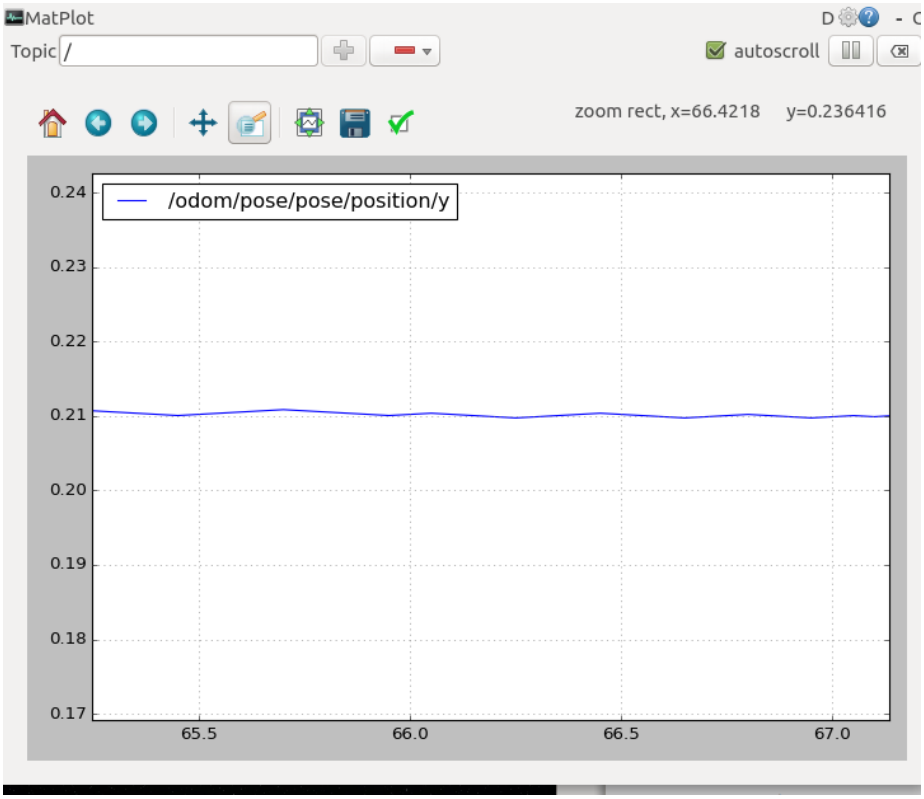
```
def do_PDC(current_y, desired_y):
    derivative = 0
    if len(odomy_arr) > 2:
        derivative = (odomy_arr[-1]-odomy_arr[-2]) / (t[-1]-t[-2])

    u = -KP *(desired_y - current_y) - KD * (derivative) + CALIBRATED_ZERO_ANGLE
    pubSteering(u)
```

For the real cat we had to use high weight values `KP = 300, KD = 100` to see acceptable results over a few meters.
Here are two plots we generated with the real car. They show the distance between `0.2 - current_y` against the time:





finally a plot of the same programm using the gazebo:



Some logs:

```
...
u= 2.9 *( 0.2 - 0.201403176746 )  + 2.6 * - 0.0153677153998 = 80.9558835035
u= 2.9 *( 0.2 - 0.201406393693 )  + 2.6 * - 0.0157934304967 = 80.9528238898
u= 2.9 *( 0.2 - 0.20140961064 )  + 2.6 * - 0.0153677153998 = 80.9587496072
u= 2.9 *( 0.2 - 0.201412827587 )  + 2.6 * - 0.0168380043108 = 80.9415971952
u= 2.9 *( 0.2 - 0.201416044534 )  + 2.6 * - 0.000204371124299 = 80.9957149158
u= 2.9 *( 0.2 - 0.201419261482 )  + 2.6 * - 0.000204653206481 = 80.9436018456
u= 2.9 *( 0.2 - 0.201422478429 )  + 2.6 * - 0.000204927768309 = 80.9037975928
u= 2.9 *( 0.2 - 0.201425695376 )  + 2.6 * - 0.0291632293682 = 80.8914564932
u= 2.9 *( 0.2 - 0.201428912323 )  + 2.6 * - 0.0389216945799 = 80.8872449446
u= 2.9 *( 0.2 - 0.20143212927 )  + 2.6 * - 0.0413891230706 = 80.8858737194
u= 2.9 *( 0.2 - 0.201435346217 )  + 2.6 * - 0.0400381427924 = 80.9007658951
```

Closing the loop: Control a car using the lidar sensor to keep its distance to a wall

We subscribed to the `/scan` topic and wrote a simple mapping function to get the distance for 80° and 100° for `dr2,dl2`.

From there values and the given constants we calculated `do2`:

```
alpha = alpha * math.pi / 180.0 #deg to rad
t = math.sqrt(dr2**2 + dl2**2 - 2 * dl2 * dr2 * math.cos(alpha))
phi2 = math.asin(dr2* math.sin(alpha) / t)
do2 = - math.sin(phi2) * dl2
```

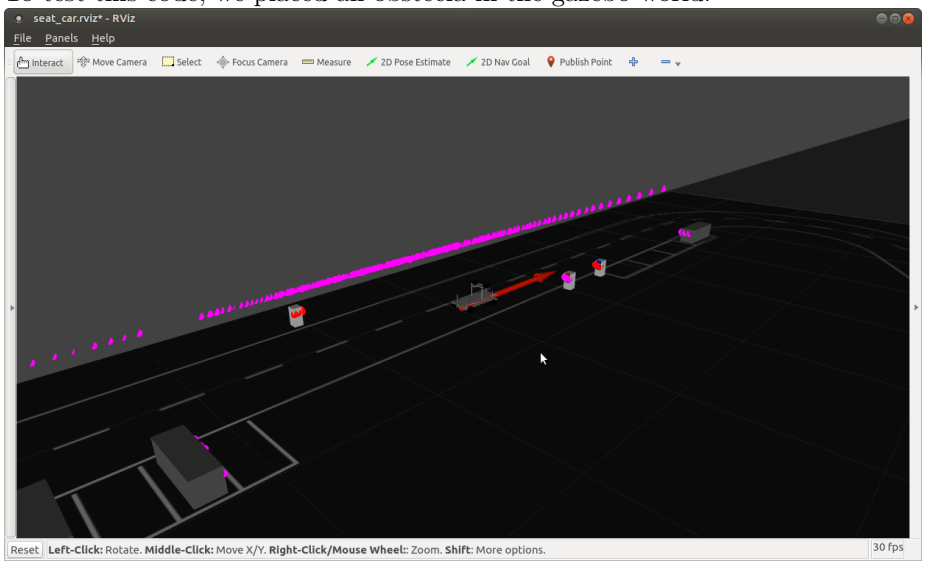
With these three values we tried to calculate the angled `theta` and `theta_stern`:

```
thetal2 = math.asin(do2 / dl2)
theta = thetal2 - alpha
cy = do2 + math.sin(theta) * s
thetaStar = math.atan2(p - cy, l) * 180 / math.pi
theta = theta * 180 / math.pi - 90
```

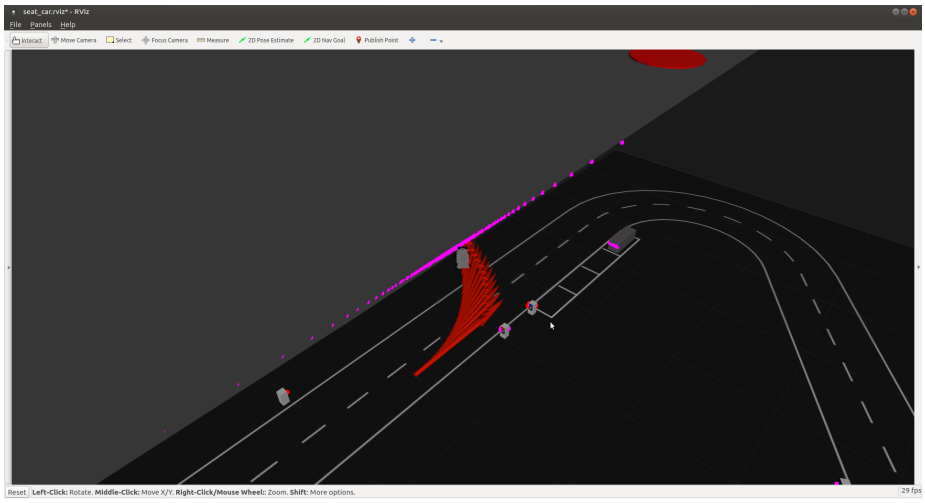
We got stable values for our `dl2,dr2,do2` but crazy values for `theta,theta_stern`. We tried to calculate the PD controller anyway:

```
deltaHeading = thetaStar - theta
derivative = 0
if len(heading_arr) > 1:
  derivative = (heading_arr[-1] - heading_arr[-2]) / (t[-1] - t[-2])
u = - KP * deltaHeading - KD * derivative + CALIBRATED_ZERO_ANGLE
```

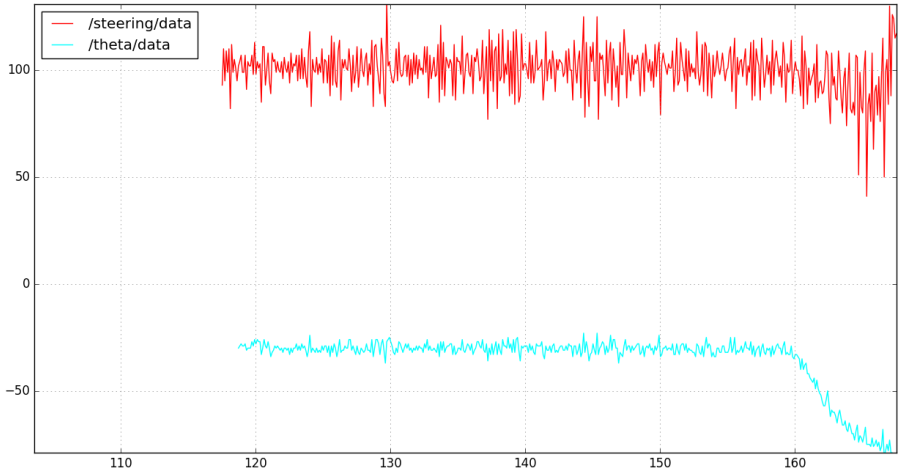
To test this code, we placed an obstecla in the gazebo world:



We get valid values for `dl2, dr2, do2` but the calculation for the steering fails:



Here is the plot of the steering angle and the theta angle. You can see the car starting af 160:



and a short log of our `dl2,dr2,do2,u` and angles:

```
dl2, do2, dr2 = 1.54757702351 1.35203625144 1.37766838074
Steering to:  -65.3189829763  from:  -47.1598292066
u =  99.0964001407

dl2, do2, dr2 = 1.54582476616 1.37078123686 1.38978350163
Steering to:  -65.7657548859  from:  -45.5757510634
u =  106.072839056

dl2, do2, dr2 = 1.5458739996 1.37152030983 1.39030718803
Steering to:  -65.7826980412  from:  -45.5199366018
u =  101.206982112

dl2, do2, dr2 = 1.55099141598 1.39573541116 1.4088037014
Steering to:  -66.3150437722  from:  -43.9003208904
u =  106.547186663

dl2, do2, dr2 = 1.55733680725 1.35097289906 1.38007211685
Steering to:  -65.2607236291  from:  -47.8769280584
u =  89.0866419516

dl2, do2, dr2 = 1.565721035 1.39512837695 1.41232419014
Steering to:  -66.252119301  from:  -45.0402025352
u =  111.443416983

dl2, do2, dr2 = 1.55085301399 1.4109273439 1.42000102997
Steering to:  -66.6502140455  from:  -42.5708167219
u =  108.341068215

dl2, do2, dr2 = 1.57046413422 1.38802262203 1.40877711773
Steering to:  -66.0773638315  from:  -45.9372685391
u =  91.1540674819

dl2, do2, dr2 = 1.56570005417 1.40422863239 1.41875243187
Steering to:  -66.4535768208  from:  -44.2955165357
u =  105.987615368

dl2, do2, dr2 = 1.56453192234 1.39986802601 1.41533517838
Steering to:  -66.36128977  from:  -44.5685078458
u =  100.350292818

dl2, do2, dr2 = 1.58244717121 1.40773680298 1.42581057549
Steering to:  -66.4760372002  from:  -45.2227212924
u =  99.7647995742

dl2, do2, dr2 = 1.59294271469 1.41472513487 1.43363761902
Steering to:  -66.5946908098  from:  -45.4070812472
u =  100.949476188

dl2, do2, dr2 = 1.58786356449 1.3861918737 1.4128446579
Steering to:  -65.9804055816  from:  -47.2369075705
u =  94.5823736518

dl2, do2, dr2 = 1.57705068588 1.41119586066 1.42674994469
Steering to:  -66.5687221881  from:  -44.5582024555
u =  109.100938386

dl2, do2, dr2 = 1.59355247021 1.43014702887 1.44466245174
Steering to:  -66.9220110893  from:  -44.2191283855
u =  102.888457342

dl2, do2, dr2 = 1.59664261341 1.41348341032 1.4338490963
Steering to:  -66.5563356226  from:  -45.7585100071
```