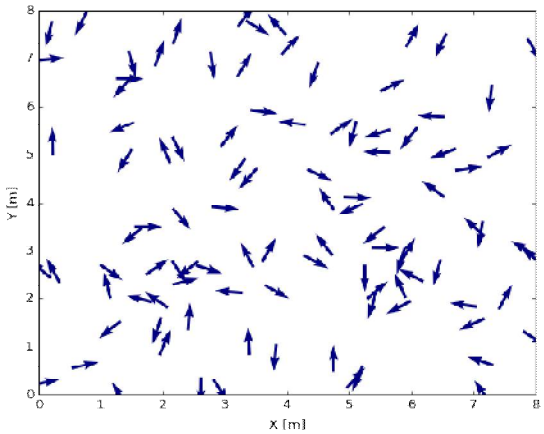
All Code can be found at: https://github.com/chbrock/robotik_ws1718

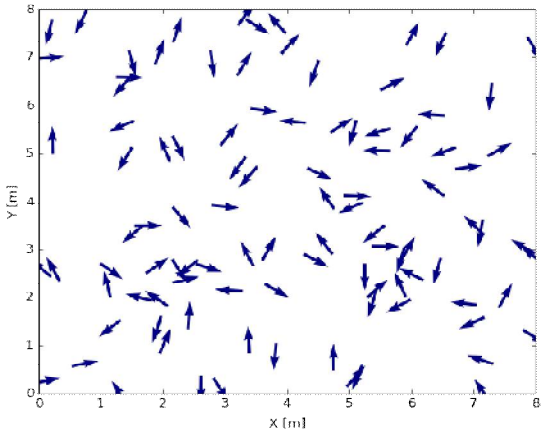
Task 1

Since the visualization in rviz proved cumbersome, we wrote our own visualization in matplotlib. The initialized particles looked like this:



Task 2

After one propagation step these look like:



Obviously the particles have not moved much….

Task 3

We used the following code snippet to calc the weight for a particle at (2, 1), yaw = 90 given the angles from position (1, 2) yaw = 1.505… using an angle variance of 0.1.

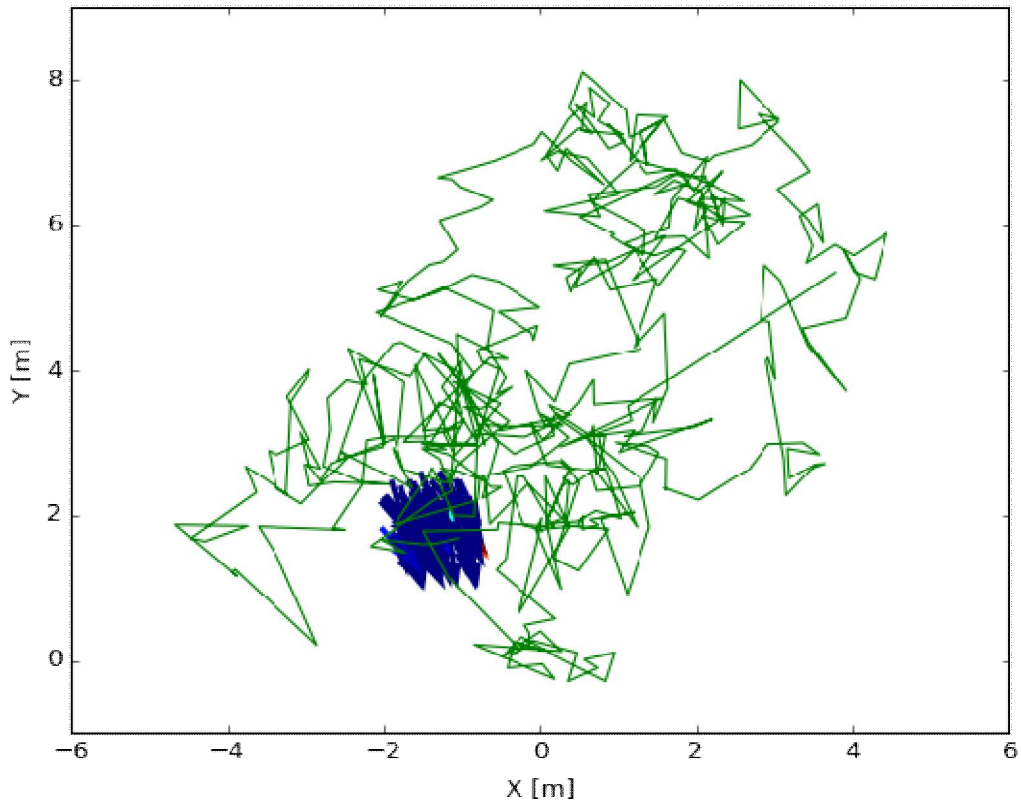
```
# doing things without the Node
rospy.init_node('part_filter', anonymous=True)
pf = PartFilter()

# overwrite the initialized particles with just one particle at (2,1) yaw = 90
pf.N = 1
pf.parts = np.array([[1, 2, np.pi/2]])

# setup a certain view in pixels
pf.pos_r = [294.60869565,  341.54347826]
pf.pos_g = [20.75675676,   361.21621622]
pf.pos_b = [215.23809524,  477.83333333]
pf.pos_p = [131.29411765,  276.72058824]

ori = np.array([0., 0.])
ang_c = np.array([rot_angle(ori, pf.pos_r),
                  rot_angle(ori, pf.pos_g),
                  rot_angle(ori, pf.pos_b),
                  rot_angle(ori, pf.pos_p)])

# get weights
print(pf.getweights(ang_c, norm=False))
```

The unormalized result is: 7.29843082e-18

Task 4

See submitted code.

Task 5



The graph shows the active converged particle field (arrows) and the trace the algorithm reported (green). Obviously that trace is wrong. This might be due to the fact that the algorithm only considers the highest probability of the particles even if that is very small (or just noise) once the particles have diverged from the real position of the car.

The positions reported by odom were never used and thus not referenced to the coordinate frame. They show the same path as they did in task 6.