9. Assignment, Introduction to Robotics WS17/18 - Ver. 1.01

Prof. Daniel Göhring, Martin Dreher, Jakob Krause, Nikolai Bobenko
Institut für Informatik, Freie Universität Berlin
Submission: online until Tuesday, 09 Jan 2018, 11:55 a.m.

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".
Submit your python code
Only one member of the group must submit the necessary files.
Do not copy solutions to other groups.
Every group must contain two people, unless granted differently.
Only submissions via KVV will be accepted.

Record a logfile (rosbag-file) in which the car is moving about 4 to 10 meters in a loop (not a circle) - i.e., start and end positions have to be the same. You shall reuse the code of recognized light bulb positions in the image.
Provide a link to your repository containing the code.

1 - Monte Carlo Particle Filter - Initialize a particle cloud. (1 point)

Initialize a set of 100 Particles randomly distributed across the field. Save the particle cloud using the topic geometry_msgs/PoseArray or visualization_msgs/MarkerArray (recommended).
and name it /mcposearray or /mcmarkerarray .
An example for how to use the marker array can be found here:
https://github.com/ipa320/accompany/blob/master/accompany_uva_msg/src/MsgToMarkerArray.cpp
Plot the distribution of your particles using rviz.

2 - Propagation with Odometry (2 points)

Apply the change of position to each particle. I.e., if the car has moved in the last time step, move each particle by that motion vector and rotate it accordingly.
Add some noise to the x,y, and angular component of each particle (different noise values for each particle) after propagating the position with odometry.

Plot the distribution of your particles before and after one time delta where the car has moved using rviz.

3 - Calculate weights for each particle - with Sensory Data (2 points)

Use the angle to light bulbs to calculate the weights for each particle as follows: exp^-[(expectedAngle - perceivedAngle)^2 / standardDeviation^2] . *Select a useful standard deviation and say, which value you used. Provide an example calculation for one particle, i.e., which expected and perceived angle did you use, what is the resulting weight.*

If you perceive multiple light bulbs, multiply the weights for each lightbulb into a final weight.

After all final weights have been calculated, normalize the final weights, so that all final weights sum up to 1 . Store the final weights, e.g., in an array.

4 - Resample, Generate a new Particle Set (3 points)

Only do the resampling, when you receive new image data.

Use low variance sampling to generate a new particle set.

- M = number of particles

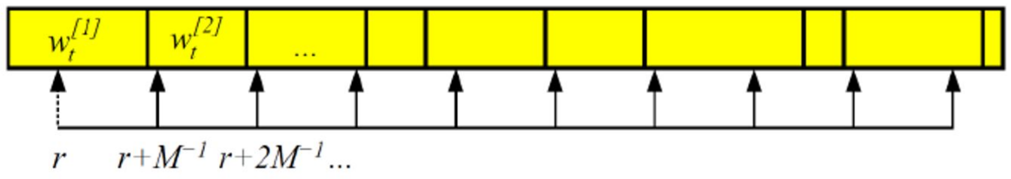

Figure 4.7 Principle of the low variance resampling procedure. We choose a random number r and then select those particles that correspond to $u = r + (m - 1) \cdot M^{-1}$ where $m = 1, \ldots, M$.

Source: Probabilistic Robotics (Thrun, Burgard, Fox), 2005.

5 - Calculate the Position (2 points)

Now create n grid cells, e.g. 50 along the x-axis times 30 along the y-axis = 1500, so that they match the field dimensions. Now calculate, how many particles fall into each grid cell and calculate the center of gravity for each cell. Use the cell with the highest number of particles, i.e., its center of gravity (for x, y, theta) for the position estimate. If multiple grid cells contain the same amount of particles, use an arbitrary one, or the one which is closest to your last position. Publish that position under mcpf_gps which is an Odometry message. *Plot the particle distribution after the distribution converged. Plot a graph of the original odom and the mcpf_gps positions.*

Hint: To calculate the average angle for a set of particles, sum up the cosines of the angles in one variable (=vc), the sum of sines in a second variable (=vs), then the average angle is arctan (vs, vc) (you probably want to use atan2(vs, vc)).