100 Points
Well Done

Robotics Lab Assignment #4

Konstantin Herwig(413869), Nicolas Marco Hahn(501703), Maximillian Holger Ehlers(474762), Jakob Arndt(399927)

January 28, 2024

Contents

1 Bayes

In this exercise, an imaginary robot has a sensor, which can detect if a door is open or closed. Our task was to calculate the probability of the door being open given a sensor measurement of $z = 42$. Also given was $P(z = 42|open) = \frac{2}{3}$, $P(z = 42|\neg open) = \frac{1}{3}$, $P(open) = \frac{3}{5}$ and that the door can either be closed or open.
Given these values we can construct two formulas with Bayes' rule 1.

$$P(open|z = 42) = \frac{P(z = 42|open)P(open)}{P(z = 42)} \tag{1}$$

$$P(\neg open|z = 42) = \frac{P(z = 42|\neg open)P(\neg open)}{P(z = 42)} \tag{2}$$

We don't know $P(z = 42)$, but by solving both formulas for $P(z = 42)$ we can create a new formula (eq. (3)).

$$P(z = 42) = \frac{P(z = 42|open)P(open)}{P(open|z = 42)} = \frac{P(z = 42|\neg open)P(\neg open)}{P(\neg open|z = 42)} \tag{3}$$

Since the door can either be opened or closed, we can say that $P(\neg open|z = 42) = 1 - P(open|z = 42)$.

$$P(z = 42) = \frac{P(z = 42|open)P(open)}{P(open|z = 42)} = \frac{P(z = 42|\neg open)P(\neg open)}{1 - P(open|z = 42)} \tag{4}$$

This equation we can solve for $P(open|z = 42)$.

$$\frac{1 - P(open|z = 42)}{P(open|z = 42)} = \frac{1}{P(open|z = 42)} - 1 = \frac{P(z = 42|\neg open)P(\neg open)}{P(z = 42|open)P(open)} \tag{5}$$

$$\frac{1}{P(open|z = 42)} = \frac{P(z = 42|\neg open)P(\neg open)}{P(z = 42|open)P(open)} + 1 \tag{6}$$

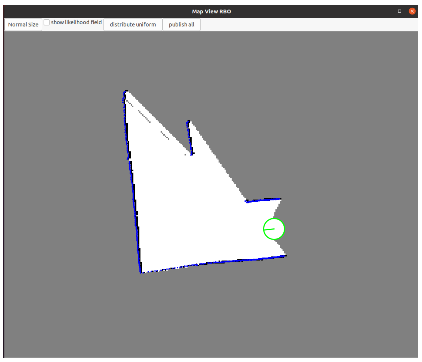
With eq. (6) we have a formula looking like this $\frac{x}{y} + 1 = \frac{1}{z}$. If we solve this for z we get $z = \frac{y}{x+y}$. With this we can construct our final formula (eq. (7)) and calculate the end result (eq. (9)).

$$P(open|z=42) = \frac{P(z=42|open)P(open)}{P(z=42|\neg open)P(\neg open) + P(z=42|open)P(open)} \tag{7}$$

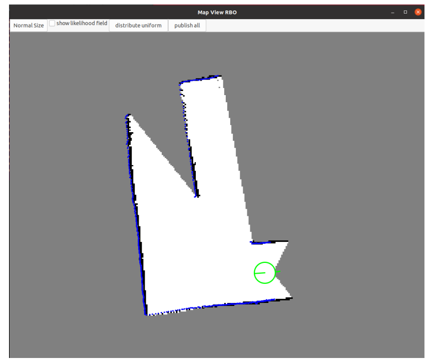
$$= \frac{\frac{2}{3}\cdot\frac{3}{5}}{\frac{1}{3}\cdot\frac{2}{5}+\frac{2}{3}\cdot\frac{3}{5}} \tag{8}$$

$$= \frac{3}{4} \tag{9}$$

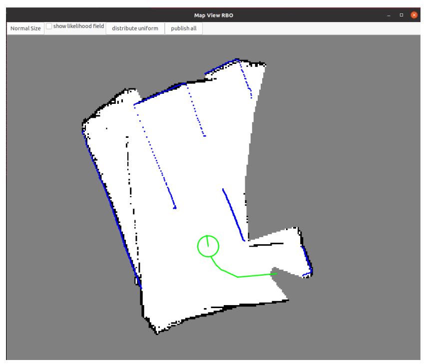
This means there is a probability of 75% that the door is open, if the sensor reports a value of $z = 42$.



(a) 2s



(b) 8s

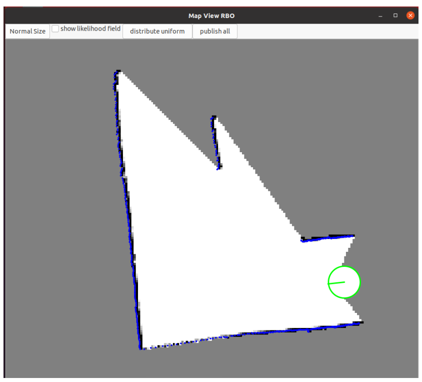


(c) 20s

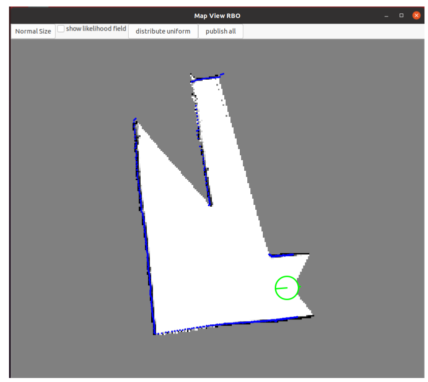
Figure 1: Resulting map when running bag 1 at different timestamps with clipping

2 Mapping with Raw Odometry

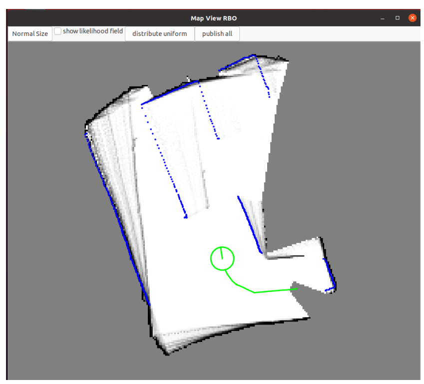
We implemented an algorithm, which generates a map based on the data given by a laser scanner. Our program uses the simple counting algorithm presented in the mapping tutorial [1]. To show our algorithm in action, we ran it twice and took screenshots of the map at $2s$, $8s$ and $20s$. The first time we ran it with clipping the resulting value of the simple counting algorithm at 0.5 (setting the map value to 0 if the value it below and to 1 if it is above 0.5). The result is shown in fig. 1. The second time we ran the algorithm without the clipping and the result is shown in fig. 2.



(a) 2s



(b) 8s



(c) 20s

Figure 2: Resulting map when running bag 1 at different timestamps without clipping

2.1 Why does the resulting map differ so much from reality?

The generated map differs a lot from the reality. The reason is that the given laser measurements and the given robot pose don't match. This is visible in fig. 2c on the top left hand corner of the drawn map. There are multiple grey lines with roughly the same length but all at a different angle to each other. This is the result of a rotating robot pose but a laser measurement which kept the same.
The difference between laser measurement and robot pose can for example be explained by slipping wheels, if the robot pose is determined by the wheel rotation.
A solution for this problem would be a more accurate robot pose generation, by for example using the laser scanner as an additional localization tool. If the reason for this inaccuracy is a very inaccurate laser scanner, maybe the laser scanner should be replaced.

3 Monte Carlo Localization: Particle Filter

3.1 Initialize Particle Distribution

Before we can use localization algorithm we have to initialize the Particles. We sample from the uniform and gaussian distribution implementations provided in the assignment for the particles x, y and θ variables. The particle weights are also normalized so that they add up to *1*, while ensuring that our particles all start out with equal weights.
Exact variable definitions can be seen in the code.

3.2 Motion model

In the first step we use a motion model that describes the possible movement of the robot. For this we use the odometry data and propagate the particles using the motion model.
Particle pose changes from:

$$\langle x, y, \theta \rangle \quad \text{to} \quad \langle x', y', \theta' \rangle \tag{10}$$

Our odometry data:

$$\begin{aligned} \delta_{\text{trans}} &= \sqrt{(x'-x)^2 + (y'-y)^2} \\ \delta_{\text{rot1}} &= \text{atan}^2\,(y'-y, x'-x) - \theta \\ \delta_{\text{rot2}} &= \theta' - \theta - \delta_{\text{rot1}} \end{aligned} \tag{11}$$

We then added noise ε with a zero mean gaussian distribution to the raw odometry data.

$$\begin{aligned} \hat{\delta}_{\text{rot1}} &= \delta_{\text{rot1}} + \varepsilon_{\alpha_1|\delta_{\text{rot2}}|+\alpha_2|\delta_{\text{trans}}|} \\ \hat{\delta}_{\text{trans}} &= \delta_{\text{trans}} + \varepsilon_{\alpha_3|\delta_{\text{trans}}|+\alpha_4|\delta_{\text{rot1}}+\delta_{\text{rot2}}|} \\ \hat{\delta}_{\text{rot2}} &= \delta_{\text{rot2}} + \varepsilon_{\alpha_1|\delta_{\text{rot2}}|+\alpha_2|\delta_{\text{trans}}|} \end{aligned} \tag{12}$$

Afterwards we can propagate each particle through the model and update their pose:

$$\begin{aligned} x' &= x + \hat{\delta}_{\text{trans}} \cos\left(\theta + \hat{\delta}_{\text{rot1}}\right) \\ y' &= y + \hat{\delta}_{\text{trans}} \sin\left(\theta + \hat{\delta}_{\text{rot1}}\right) \\ \theta' &= \theta + \hat{\delta}_{\text{rot1}} + \hat{\delta}_{\text{rot2}} \end{aligned} \tag{13}$$

3.3 Measurement model

For our measurement model we have used the *Likelihood Fields for Range Finders* described in the book ("Probabilistic Robotics", Thun et. al. page 169 pp.). We access and filter our laser data using the documentation provided at `https://docs.ros.org/en/noetic/api/sensor_msgs/html/msg/LaserScan.html`.
Notable implementation details are:

1. the usage of the map resolution for the coordinates to access our likelihood field
2. laser data can point to coordinates outside of our map so we have implemented a weight penalty for such cases, making sure that scans within the boundary are prioritized
3. weight normalization at the end of the function

For the a screenshot of the likelihood field in log space see Figure 3

3.4 Resampling

See code.

3.5 Global Localization

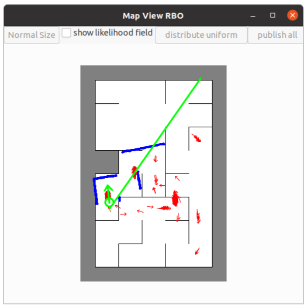
3.5.1 Uniform

See Figure 4
The big position jumps at the beginning of the localization can be explained by increasing confidence of the best hypothesis over the runtime. At the start, the best hypothesis might simply be inaccurate which will be corrected over time.

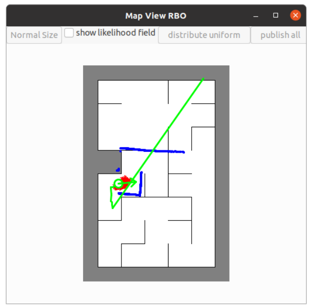
3.5.2 Tracking

See Figure 5

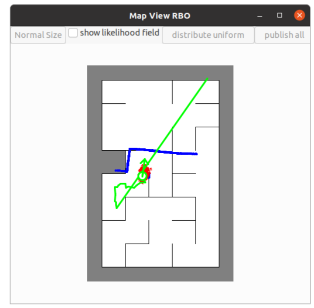
Figure 3: Likelihood field in log space



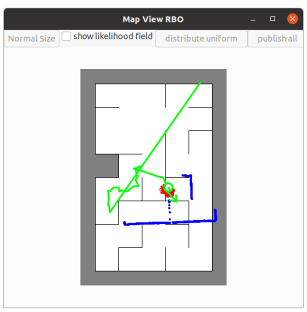
(a) 5s



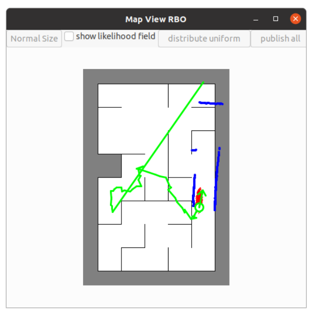
(b) 10s



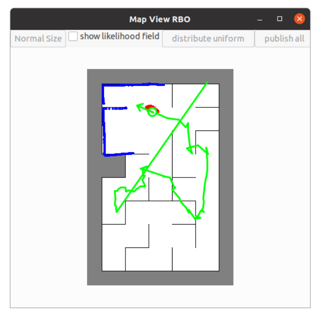
(c) 14s



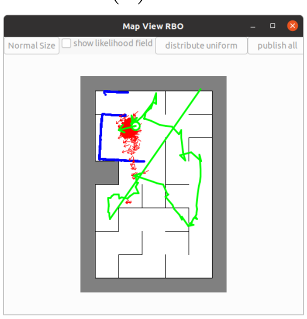
(d) 20s



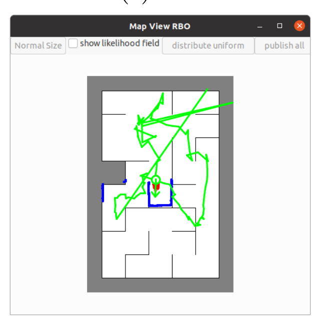
(e) 30s



(f) 42s

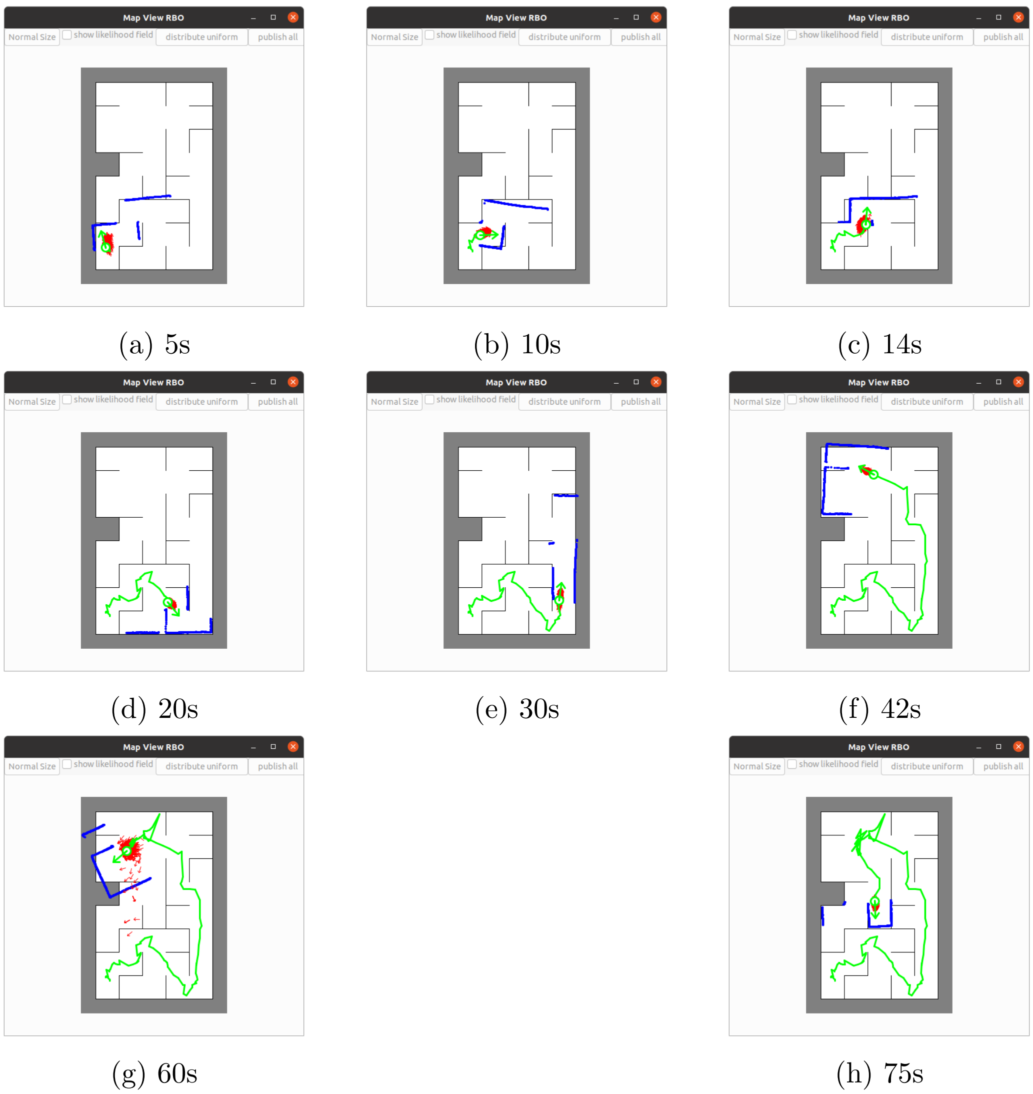


(g) 60s



(h) 75s

Figure 4: Resulting map when running with uniform distribution



(a) 5s (b) 10s (c) 14s

(d) 20s (e) 30s (f) 42s

(g) 60s (h) 75s

Figure 5: Resulting map when running with gaussian distribution

4 Appendix

Student Name	A1	B1	B2	B3	C1	C2	C3	C4	C5	C6
Konstantin Herwig	x	x	x	x	x	x	x	x	x	x
Nicolas Marco Hahn	x	x	x	x	x	x	x	x	x	x
Maximillian Holger Ehlers	x	-	-	-	x	x	x	x	x	x
Jakob Arndt	x	x	x	x	x	x	-	-	-	-

Table 1: Tasks done by which student

References

[1] Manuel Baum. "Mapping Tutorial". In: *Robotics WiSe23/24* (). URL: https://isis.tu-berlin.de/pluginfile.php/2894507/mod_folder/content/0/R04.T.Mapping.pdf?forcedownload=1.