Monte Carlo localization

Efficient Position Estimation for Mobile Robots

Jan-Tarek Butt - Matr.-Nr. 7328242

University of Hamburg
Faculty of Mathematics, Informatics and Natural Sciences
Department of Informatics
Technical Aspects of Multimodal Systems

11. June 2020

Introduction into MCL

A way of locating, orientating and location tracking in a known environment.

- Date of publication: 1999
- Authors: Dieter Fox, Wolfram Burgardy, Frank Dellaert, Sebastian Thrun
- Paper-title: Monte Carlo Localization: Efficient Position Estimation for Mobile Robots
- Place of Publication: Appeared in Proc. of the Sixteenth National Conference on Artificial Intelligence (AAAI-99)

Outline

General about localization

Position Tracking

- Last robot position is known.

Global localization

- Robot can be anywhere thus the initial belief is uniformly distributed.
- Map needs to be known. Otherwise no localisation is possible.

Grid-based Markov Localisation

- Can start with any initial belief state.
- Algorithm is a combination of two steps:
  - a prediction step
  - a correction step

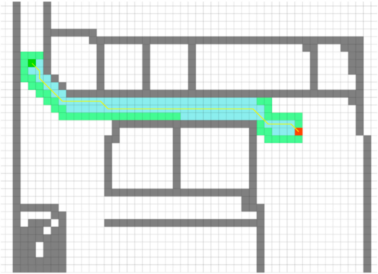

Fig. 1: 2D example of grid-base Markov Localisation

Robot motion model (prediction)

- Previous Belief $bel(l')$
- Movement action: a

$$\overline{bel}(l) \leftarrow \int P(l|l', a) bel(l') dl' \tag{1}$$

For every possible previous location (l') consider:

- Probability that input a brought the robot from l' to l
- Probability of the robot having been there

Robot sensor-update (correction)

Uses Bayes rule to update belief based on:

- Predicted belief: $\overline{bel}(l')$
- External sensor measurements: s

$$bel(l) = \alpha p(s|l, M)\overline{bel}(l') \tag{2}$$

Where $p(s|l, M)$ is the probabilistic measurement model.

Grid-based Markov Localisation issues

Problems are:

- The smaller the grid size the higher the memory consumption.
- The smaller the grid size the more computing power is necessary.
- Localization resolution can not be higher than the grid size.

Goal of the Paper

Find a more efficient way to localize in a known environment

- More accurate solution of localisation ...
- Less computationally cumbersome ...
- Less memory consumption ...

... as compared to previous (grid-based) approaches.

Monte Carlo Localisation

How does MCL work?

Particles are described as: tuples of x, y, θ where θ is the orientation.

Similar to the previous Markov Localisation approaches, MCL also works with Belief in particles and their positions. But MCL randomly distributes them instead of placing them on a rigid/static grid. This use of randomness is the Monte Carlo in MCL

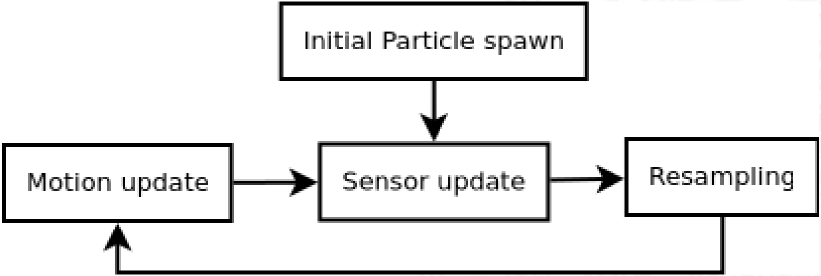

Fig. 2: flowchart of MCL steps

Monte Carlo Localisation - Visualisation

- Red triangles are simulated particles
- The green triangle is the actual robot

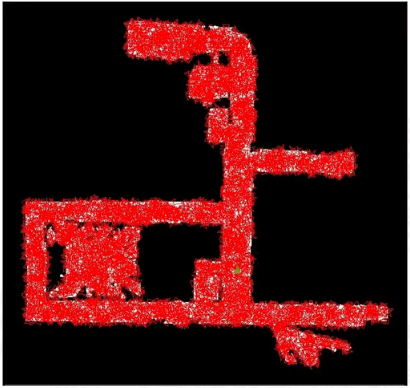

Fig. 3: Simulation of a robot moving in a building

Steps - initial particle spawn

Fig. 4: Upper part shows the illustrated environment. Lower part the uniformly distributed amount of particles

The algorithm initializes with a random uniform distribution of particles. The robot considers itself to be equally likely at any point in space along the illustration, even though it is physically at the first door.

Steps - sensor update (correction)

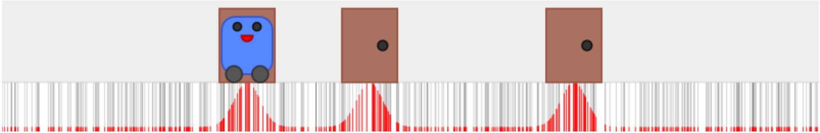

Fig. 5: The red lines are the new probability

Sensor update: the robot detects a door. It assigns a weight to each of the particles. The particles which are likely to give this sensor reading receive a higher weight. Notice the number of particles is always the same.

Steps - resampling

Fig. 6: Resampled set of new particles

Resampling: Particles with low belief get respawned around particles with high belief.
The Robot now believes it is at one of the three doors.

Steps - moving

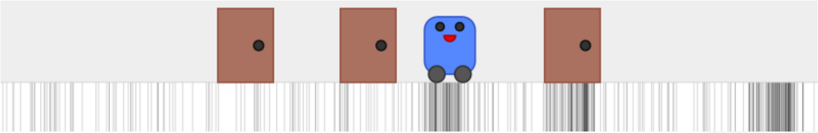

Fig. 7: Movement update all particles move in the same direction as the robot

Motion update: If the robot drives right only correct particles will remain as the particles currently at the third door and at the very right will get a low believe after the next sensor update.

The robot has been localized.

Comparison with others

Grid based Markov Localisation

- \- Memory consumption.
- \- Computing power.
- \- Localization resolution.

Monte Carlo Localisation

- $+$ Memory consumption.
- $+$ Computing power.
- $++$ Localization resolution.

Variances of MCL

Kidnapping problem
- Take the robot and place it some where else. Wrong estimated position problem
- recover in case of false positive position.

- Solution is an addition of random noise on new particles while resampling.

Adaptive Monte Carlo localization (AMCL) use variable amount of particles depending on the believe. The higher the believe the less particles unnecessary to maintain tracking.

References

(Fig. 3): https://www.youtube.com/watch?v=DZT-zNj61Jg
https://rse-lab.cs.washington.edu/papers/sampling-aaai-99.pdf
https://de.wikipedia.org/wiki/Monte-Carlo-Simulation
https://en.wikipedia.org/wiki/Monte_Carlo_localization
(Fig. 1): Paper: UAV Autonomous Localization Using Macro-Features Matching with a CAD Model (Fig. 2): self made
(Fig. 4 to 7):
https://en.wikipedia.org/wiki/Monte_Carlo_localization Book:
http://probabilistic-robotics.org/