Localization and Path Planning

Applications for Mobile Robot Navigation

Massimo Innocentini

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

17. May 2018

Outline

Navigation

- Localization and Path planning are competences needed for navigation

- Mobile robot navigations goals:
  - Reach goal destination.
  - Avoid obstacles in the way.

"navigation encompasses the ability of the robot to act based on its knowledge and sensor values so as to reach its goal positions as efficiently and as reliably as possible"[1]

Localization

The question localization is trying to answer: Where am I ?

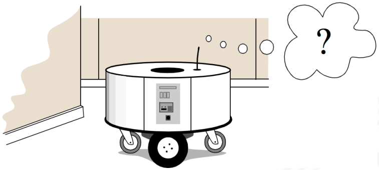


Figure 1: Where am I?[1]

- Robot must determine its position w.r.t a map of the environment.
- We want to do that accurately.

Localization taxonomy

The localization problem can be divided into three types:[2]

- Relative localization
- Absolute localization
  - Kidnapped robot problem

Another characterisation of localization is the type of environment:

- Static
- Dynamic

Path planning

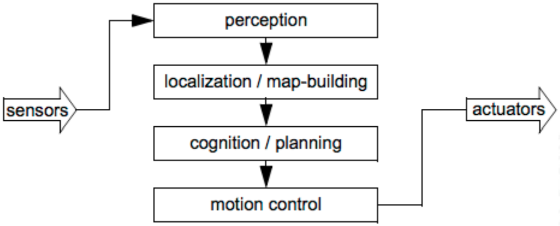


Figure 2: Map-based architecture navigation[1]

- Robot has map and its location in it.
- Need to find a collision-free trajectory to reach the goal location.

Motivation

There are other possible approaches, however they have limitations:[1]

- Behaviour based approach
  - Not flexible
  - Too many assumptions
- Dead reckoning
  - Sensors / Effectors noise
  - Information is not enough
- GPS
  - Expensive
  - Accuracy
  - Coverage
  - Other objetcs

State estimation

"Probabilistic robotics is the idea of estimating state from sensor data"[2]

- We won't know exactly every detail from the sensors:
  - Noise
  - Partial information
- A `state` is defined as the collection of all aspects about a robot and the environment which can affect the future.[6]
  - Robot pose / speed
  - Walls
  - Obstacles
  - People

Bayes rule

Applying Bayesian statistics and probability distributions to estimate the robot state.[2]

Bayes rule:

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)}$$

- $p(x) =$ *Prior probability distribution*
- $p(y) =$ *Data*
- $p(x|y) =$ *Posterior probability distribution*

Bayes filter

Let's introduce some terminology:

- *belief* = Possible state
- $\mathrm{x_t}$ = *State*
- $\mathrm{z_t}$ = Measurement data coming from environment
- $\mathrm{u_t}$ = Control data eg. Odometers data

```
1. Algorithm Bayes Filter($bel(x_{t-1}), u_t, z_t$):
2. for all x do:
       $\overline{bel}(x_t) = \int p(x_t|u_t, x_{t-1}) bel(x_{t-1}) dx$
       $bel(x_t) = p(z_t|x_t)\overline{bel}(x_t)$
3. endfor
4. return $bel(x_t)$
```

Bayes filter cont.

The Bayes filter is recursive and divided into two steps:[2]

- Prediction step:

$$\overline{bel}(x_t) = \int p(x_t|u_{t-1})bel(x_{t-1})dx$$

- Measurement update:

$$bel(x_t) = p(z_t|x_t)\overline{bel}(x_t)$$

A $bel(x_0)$ for $t = 0$ is needed, ideally useful prior knowledge.

Map representation

Map representation is another pillar in mobile robots navigation.

Close connection between:[1]

- Map precision $\rightarrow$ Goal precision
- Map precision $\rightarrow$ Sensors precision
- Map complexity $\rightarrow$ Reasoning computational complexity

- Maps have a large impact on both localization and path planning.

- Robot operation context crucial for map representation

Map representation types

Possible representations:

- Continuos representation
  - 2D representation
  - Line extraction
- Decomposed representation
  - Cell decompisition
  - Occupancy grid map

Continuous representation is more "faithful" but harder to reason with.

Abstract representation may lack feature but decrease complexity of reasoning.

Markov localization

- Probabilistic localization algorithms are all variants of the Bayes filter.

- Markov Localization:[2]
  - Solve all three localization problems in static environments.
  - Need a map.
  - Estimate each possible robot position.

```
1. Algorithm Markov Localization($bel(x_{t-1}), u_t, z_t, m$):
2. for all x do:
       $\overline{bel}(x_t) = \int p(x_t|u_t, x_{t-1}, m) bel(x_{t-1}) dx$
       $bel(x_t) = p(z_t|x_t, m)\overline{bel}(x_t)$
3. endfor
4. return $bel(x_t)$
```

Markov Localization Example

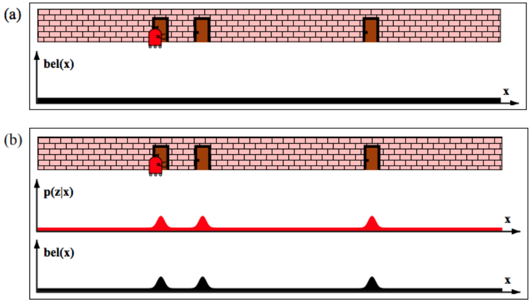


Figure 3: Hallway example[2]

Initial uniform distribution, robot could be anywhere.

The robots uses its sensor to recognize it's next to a door.

Markov Localization Example

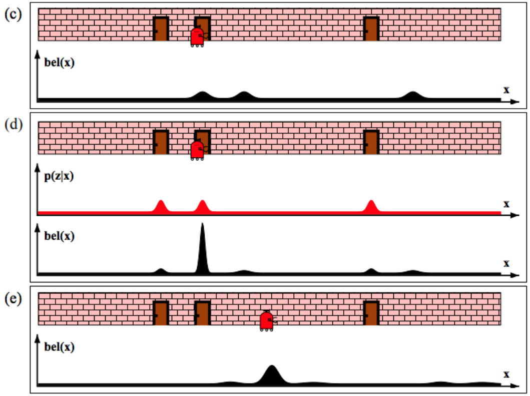


Figure 4: Hallway example continued[2]

The robot keeps moving and integrate the motion to its belief.

Again the correction step is executed with the data from the sensors.

Kalman Filter

Markov Localization is useful but:

- Computationally heavy.
- Markov assumption.

- Often Markov Localization is inefficient and not really applicable.
- More efficient solution: Kalman filter for localization purposes.
- Localization as a sensor fusion problem.

Extended Kalman Filter

- The general Kalman filter is not applicable: linear state transition / measurement assumption
- The belief calculated is only an approximation of the true belief.
- EKF Localization is a special case of Markov Localization.

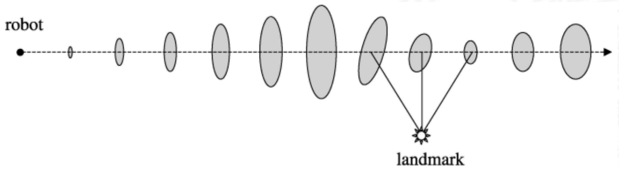


Figure 5: EKF Localization using a landmark feature map[2]

Particle Filter

- Non parametric implementation of Bayes filter.
- Estimate subset of possible position.
- Distribution represented by random samples (Particles).
- Easy to implement
- Most versatile, still few issues:
  - Number of particles
  - Random resampling error
  - Hard to match proposed - target distribution
  - Particle deprivation problem

SLAM

Simultaneous Localization and Mapping

- No access to map.
- No access to own pose.
- More complex then previous problem.
- Both continuous and discrete components.
- Two problems type:[2]
  1. *online* SLAM
  2. *full* SLAM

EKF Case Study Introduction

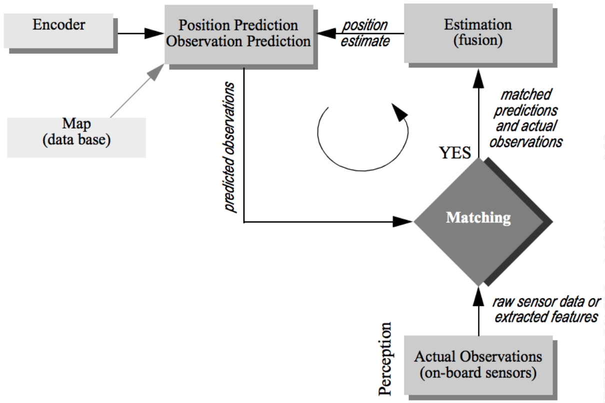
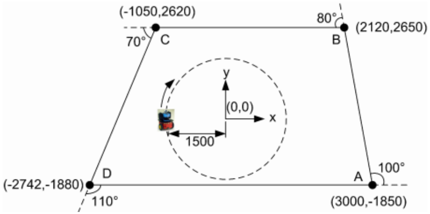


Figure 6: Kalman filter localization model[1]

EKF Case Study Methodology

- Comparing odometry based localization vs EKF localization.
- Robot moving in circle in a squared map.



1. Feature Extraction
   - Feature map representation
2. Motion model
   - Odometry model
3. EKF Application

Figure 7: Experimental Environment model[4]

EKF Case Study Result

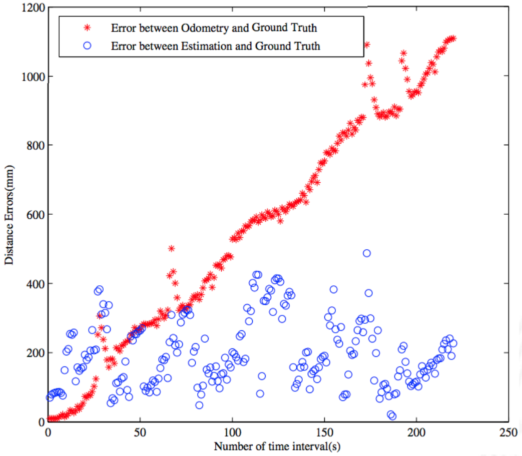


Figure 8: Experiment result data[4]

- After running the experiment the EKF showed to be more reliable.
- Euclidean distance used as accuracy reference.

$$DEOG_i = \sqrt{(x_{o,i} - x_{g,i})^2 + (y_{o,i} - y_{g,i})^2}$$

$$DEEG_i = \sqrt{(x_{e,i} - x_{g,i})^2 + (y_{e,i} - y_{g,i})^2}$$

Path Planning Techniques

- Strong connection with motion control.
- Problem simplified due to less degrees of freedom.
- Configuration space is greatly reduced.

$$P = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}$$

Common assumptions made during execution:[3]

1. A map is provided.
2. Robot know its location in the map.
3. Obstacle avoidance competence.

Graph Theory Algorithms

- Environment model has to match chosen search algorithm requirements.
- Decompose the environment into a graph.
- With a graph planning is reduced to shortest path problem.

Common shortest path search algorithm:

- Dijkstra
- A*

Dijkstra

- Dijkstra one of the oldest searching algorithm.
- Developed around 1956
- Dijkistra is complete, always consider node with lowest cost.
- Obviously highly inefficient in large spaces.

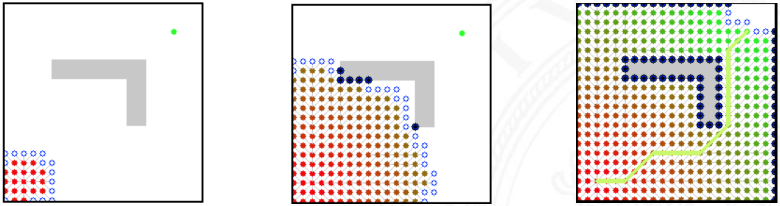


Figure 9: Short sequence of Dijkstra applied[1]

[1] https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm

A*

- An improved version of Dijkstra is called A*.
- Developed around 1986, applied on Shakey.
- User prior knowledge to improve performance.
- Eg. estimating the distance between a node and the goal node.

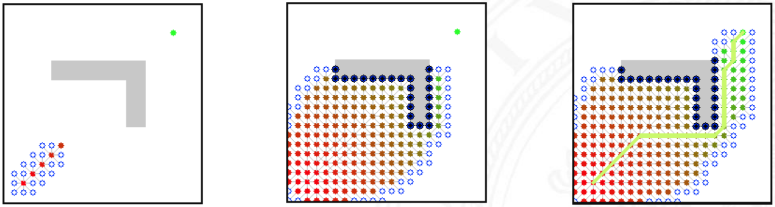


Figure 10: Short sequence of A* applied[2]

[2] `https://en.wikipedia.org/wiki/A*_search_algorithm`

Sampling-based Path Planning

- Both Dijkstra and A* are inefficient in large search space.
- A possible solution $\rightarrow$Sampling-based algorithm.
- Sampling-based path planners are *probabilistic complete*
  - Rapidly-exploring Random Trees (RRT)
  - Probabilistic Road Map (PRM)
- As time goes to infinity, the probability to find a path tends to 1.

Rapidly-exploring Random Trees - RRT

- Grow a single tree from the robot starting position until one of the branches reaches the goal.

- Basic algorithm steps:[3]

```
1. Tree=Init(X,start);
2. WHILE ElapsedTime() < t AND NoGoalFound(G) DO
       newpoint = StateToExpandFrom(Tree);
       newsegment = CreatePathToTree(newpoint);
       IF ChooseToAdd(newsegment) THEN
           Tree=Insert(Tree,newsegment);
       ENDIF
3. ENDWHILE
4. return Tree
```

Rapidly-exploring Random Trees - RRT

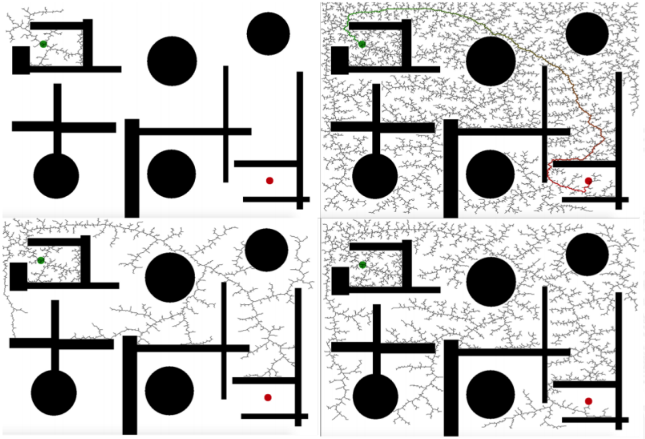


Figure 11: RRT application in 2D search space[3]

Extendend RRT Case Study Methodology

- RRT applications in real-time navigation scenario.
- Improvements:[5]
  - KD-Tree to speed up nearest neighbour lookup
  - Better encoding for tree's points
  - Using old iteration to aid new ones: Waypoint cache

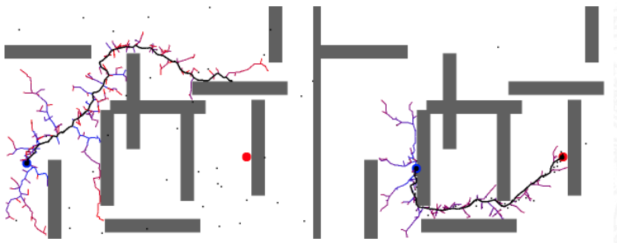


Figure 12: ERRT simulation[5]

ERRT Case Study Results

Results of experiment:[5]

- Tested on RoboCup small size.
- Initial results:

ERRT	Reactive
0.8m/s	1.2m/s.

- Improved replanning by using a straight path when possible.
- With improvements:

ERRT	Reactive
1.7m/s	1.2m/s.

RRT Considerations

- Basic RRT is a good solution for path planning.
- It can be applied for real-time scenarios.
- The basic approach has few downsides:
  - Unnecessary motion
  - Single-query
  - Not optimal solution
- However there are already different improved approcahes ERRT, RRT*, Particle RRT.

Conclusion

- We have seen what navigation entitles.
- Localization is a crucial aspect of navigation.
- There is not certain robot's position but we can estimate it.
- Robot view of the world is correlated to the path-planning algorithm.
- `Complete` search algorithm are easy to implement but inefficient in most cases.
- There is not one true best path-algorithm.

`In conclusion both localization and path-planning are crucial aspects in order to build a true autonomous robot.`

Bibliography

1. Siegwart, R. and I.R. Nourbakhsh, Introduction to Autonomous Mobile Robots. 2004
2. Thrun, Sebastian and Burgard, Wolfram and Fox, Dieter, Probabilistic Robotics. 2005
3. Nikolaus Correll, Introduction to Autonomous Robots, v1.7. 2016
4. L. Chen, H. Hu and K.McDonal-Maier, EKF based Mobile Robot Localization. 2012
5. J. Bruce and M. Veloso, Real-Time Randomized Path Planning for Robot Navigation. 2002