# Localization techniques for autonomous mobile robots

João Silva, Nuno Lau, António J. R. Neves

*Abstract* – **Mobile autonomous robotics is nowadays an area of study much addressed by research teams worldwide. One of the main challenges to create robots that can be really autonomous is the self-localization problem. For a robot to be able to plan a motion and move in a useful way, it should know where it is in the environment. Otherwise, it would just move randomly, probably not being useful at all. Then, there is also the case when one does not have a map to provide to the robot. In those situations, the robot should be able to build a map and localize itself relatively to it on runtime. This document aims to provide a brief presentation of these problems and some of the currently used solutions.**

## I. INTRODUCTION

Self-localization and mapping are classic problems of intelligent mobile robotics, over which research is still extremely active. These are part of the more general challenge of defining, managing and updating the robot internal world model. Sensor and information fusion techniques are widely used for these tasks. Generally, robots have access to partial and uncertain information through a set of multi-modal sensors. In dynamic environments, information fusion (of historical information and information coming from different sensors) is essential for the world model to be as precise as possible. Information fusion for localization is usually addressed through the use of probabilistic techniques such as Kalman or particle filters, sometimes conjugated with maximum likelihood techniques.

The integration of information over time in order to filter sensor noise is essential to get better estimates. This type of integration may be performed using Kalman filter based approaches, Monte-Carlo methods or Markov approaches. Generally, Monte-Carlo [1] approaches have better performance in cases where great discontinuities of the output values are expected, as the assumption of Gaussian probability density functions of the Kalman filter [2] is usually less accurate. However, Kalman filtering is a very effective method if the assumptions of Gaussian noise can be met and the system can be linearized. Other common approaches are the use of the Extended and Unscented Kalman filters [3], which are prepared to deal with non-linear systems at the cost of more computational weight.

When working with mobile autonomous robots, there are typically two scenarios. The environment can be known or partially known and the robot needs to localize itself, or the environment is unknown and the robot needs to build the map as it runs. This is addressed as Simultaneous Localization And Mapping (SLAM) and it is another common application of sensor fusion techniques [4], [5].

This document presents the main issues of the localization problem in Section II. A brief summary of some of the most commonly used localization algorithms is presented in Section III. Section IV briefly presents the SLAM problem and common approaches to the mapping and SLAM forms. Some remarks are presented in Section V.

## II. LOCALIZATION PROBLEM

The problem of mobile robots localization is to identify where a robot is, given a map of the environment around it. The localization of a robot is usually defined as a pose, which contains a position (given in some coordinate system) and an orientation (relative to the defined coordinate system). When a robot is running and performing localization by its own means, it is not sure where it really is and how it really is oriented. The pose that it keeps is thus called *beliefs*, as the robot believes it has a particular pose, but there is no guarantee that it is really correct; it is an estimate. This belief can be presented as a probability distribution function, $bel(x)$, where $x$ is the pose of the robot, whichever coordinates are used (it can be a 1D, 2D or 3D situation, both for position and orientation).

At some moments of the run, the robot gets information from sensors that are at its disposal. These sensors can get information about the surroundings and provide information with some degree of accuracy about the pose at that instant. A robot can be equipped with a variety of sensors to help localization purposes, but this will not be subject to analysis. Let us just consider that the sensors provide measurements $z$ that are also not 100% accurate, they have some noise associated. The observations are represented by an observation model written as $p(z|x)$, which is a probability function.

Thus, the localization problem is mostly a probabilistic problem. Robots must have models for their movement (*motion models*) that are capable of providing the beliefs. These beliefs can then be reinforced or not by sensor measurements.

The localization problems are usually divided considering different aspects that are not equally difficult to solve. There are four main aspects to consider [4]:

### A. *Local versus global localization*

This characterizes the problem by the type of knowledge available initially and at run-time. Three different problems are distinguished, each with increasing difficulty.

- **Position tracking**. Position tracking assumes that the initial pose of the robot is known. That means that the localization algorithm has to estimate the new pose based on the last one, which usually has a small error that can be accommodated in the model with good results.
- **Global localization**. In this case, the initial pose is unknown. The robot is placed somewhere in the environment and thus no assumptions on the limits of the pose error can be made. Global localization includes the position tracking problem.
- **Kidnapped robot problem**. This is a variant of global localization with an added difficulty: during operation, the robot can be kidnapped and teleported to other location. The robot might believe that it knows where it is while it does not, leading to subsequent wrong pose estimations due to false initial pose knowledge. This leads to a new question of how can the pose estimation be validated and how can a wrong pose be detected. In the global localization problem, this question is not relevant, as the robot knows that it does not know where it is.

### B. *Static versus dynamic environments*

The environment dynamics also causes a substantial impact on localization difficulty. The environment is typically classified in two classes.

- **Static environment**. In this kind of environments, the robot is the only element with motion, meaning the only state variable is the pose of the robot. All the other objects in the environment remain at the same location forever.
- **Dynamic environment**. These environments have objects other than the robot whose position and configuration may change over time. Some examples of more significant changes are people, movable furniture, doors, or even light conditions (daylight, night).

Working with dynamic environments creates more difficulties than working with static ones. There are two main approaches for accommodating dynamics: one is to include the dynamic entities in the model state vector. This approach also maps the environment, but it comes with a high burden of computational and model complexity. Another approach is to filter the sensor data to correct the damage caused by unmodeled dynamics [4].

### C. *Passive versus active approaches*

This aspect of localization problem characterization pertains to the fact of whether or not the localization algorithm can influence the control over the robot.

- **Passive localization**. The localization module only observes as the robot operates. The robot is controlled by other means, usually performing the tasks it is assigned to.
- **Active localization**. On active approaches, the localization algorithms have direct influence over the robot's control and attempt to move it to a more favorable position to reduce the error and obtain a better pose.

Active approaches tend to be easier to deal with than passive ones and tipically yield better results as well. An example is a robot located in a symmetric corridor where the global localization can easily enter an ambiguity state (Fig. 1). The local symmetry makes it impossible to localize the robot while in the corridor. It will only be able to eliminate the ambiguity and determine its pose if it moves into a room (or if it gains access to some other disambiguation source, like a true heading, for instance).

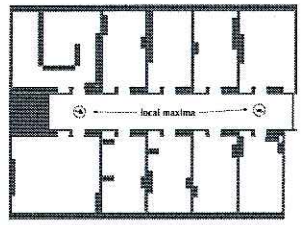



Fig. 1 - Example situation with two possible ambiguous poses. The robot would need extra information or to enter one of the rooms to determine its true location. Image from [4].

However, in practice, an active localization technique tends to be insufficient if applied on its own. Most of the times, the robot has to be able to execute other tasks besides localization. A common approach is to merge the localization goals with the task goals. An example would be a robot moving from a point A to B taking a longer path, but ensuring that the robot kept itself well localized along all the way.

### D. *Single-robot versus multi-robot*

The fourth aspect of the localization problem is related to the number of robots involved

- **Single-robot**. This is the most typically studied approach, dealing with only one robot. It is convenient that all the data comes from the same platform, and no communication issues have to be dealt with.
- **Multi-robot**. When working with a team of robots, the problem can be addressed as several single-robot localization problems and be solved in the same manner. However, if the robots have the ability to detect each other, one robot's beliefs can be used to validate other robot's beliefs if the relative location of both is available.

## III. LOCALIZATION ALGORITHMS

As localization is a probabilistic problem, most of the algorithms for mobile robot localization are based on Bayesian rules. Some algorithms will be briefly presented in this section.

### A. Markov localization

Markov localization is the straightforward application of a Bayes filter to the localization problem. It requires a map as input for the measurement model and often the map is also incorporated in the motion model. It mainly transforms a probabilistic belief at time *t-1* into a belief at time *t*. Being a probabilistic belief at each instant, it maintains the probability for every possible pose on the state space.

Markov localization addresses the global localization problem, tracking problem and the kidnapped robot problem in static environments.

Consider a scenario of a hallway with three identical doors and the robot is only able to move along the hallway without rotating. It is a one-dimensional example (Fig. 2).

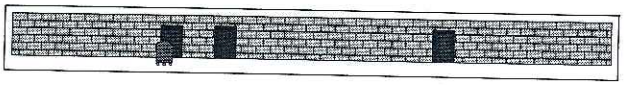


Fig. 2 - One-dimensional scenario for robot localization. Image from [4]

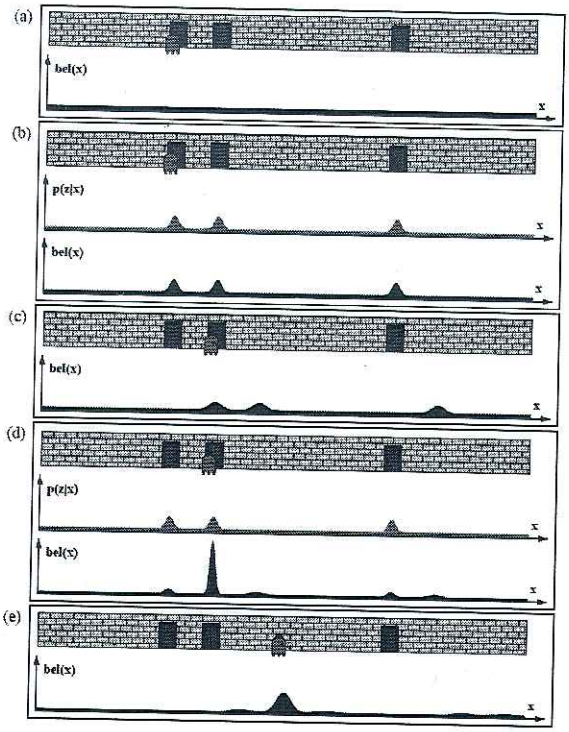



Fig. 3 - Illustration of Markov localization algorithm. In each picture, the belief $bel(x)$ function is represented. In (b) and (d), also the observation model $p(z_t|x_t)$ is represented, describing the probability of observing a door at the different locations in the hallway. Image from [4].

Figure 3 illustrates this one-dimensional example (presented in [4]), where a robot moves along a corridor with three identical doors. In an initial state, the belief $bel(x) = bel(x_0)$ is uniform for all poses along the corridor (Fig. 3.(a)).

In Fig. 3.(b), the robot queries its sensors (modeled by a function $p(z|x)$) and detects that it is in front of a door. The belief is updated with that information, resulting in the belief in the same image.

In a third moment, the robot moves right, and the model convolution results in the belief depicted in Fig. 3.(c), flattened by the assumptions of the motion model uncertainty.

A new sensor query is made in Fig. 3.(d) and the resulting belief now has a well defined peak focused on the correct pose. At this point, the robot is quite confident that it has localized itself. Figure 3.(e) illustrates the robot's belief after a new move to the right.

This illustration refers to the global localization problem. In a position tracking problem, the initial position would be known and thus the initial state would be something like the belief in Fig. 3.(d).

A Markov localization approach has been explored in robotic museum tour guides [6]. The work described in [6] was tailored for dynamic environments and was developed to support both global localization and kidnapped robot problems (recovery from location failures).

### B. Kalman filter localization

Kalman filter localization typically requires that the starting position of the robot is known, thus, in its essence, it addresses the position tracking problem. Contrary to Markov localization, it maintains a belief $bel(x_t)$ that is a Gaussian function and thus can be represented by its mean and covariance.

The Kalman filter is a two step process:

First a forecast of the output is made based on the linear evolution of the noisy dynamic system, an *a priori* estimate.

Then, on a second phase, a measurement based on the sensors is combined with the forecast in order to produce a final *a posteriori* probabilistic estimate of the pose [7]. If no sensor readings are available, the belief generated by the motion model tends to degrade, because it relies only on the forecasts of the pose until a new measure can help correct it.

Kalman filter is based on linear dynamical systems discretized in time. It is assumed that both the system and the measures are affected by *White Gaussian* noise, meaning the noise is not correlated in time, and thus we can assume that at each discrete time, the noise affecting the system and measures are independent of past or future values.

In the corridor example of [4] (depicted in Fig. 4), it is assumed that the map is represented by a collection of features (where each one is identified by a *correspon-*

*dence variable*) and each feature identifier is known, thus, the doors of the hallway have now a unique *correspondence variable* (1, 2 and 3). The need for these constraints is that the filter works on the assumption of Gaussian measurements and for this requirement to be met, each door must be unequivocally identified by the observation model $p(z|x)$ .

A second assumption, needed for the algorithm, is that the initial position is relatively well known (the initial belief $bel(x_0)$ is represented by the Gaussian distribution shown in Fig. 4.(a), near door 1 and with a Gaussian uncertainty).

As the robot moves right (Fig. 4.(b)), its belief is convolved with the Gaussian motion model, resulting in an increase of the Gaussian width, as uncertainty increases. In another instant, the robot detects that it is in front of door number 2. In Fig. 4.(c), the observation function $p(z|x)$ is used to update the estimated pose, and the resulting belief is presented in the same picture. The variance of the resulting belief is smaller than both the previous belief and measurement variances, thus integrating two independent estimates should make the robot more certain than each of the estimations separately.

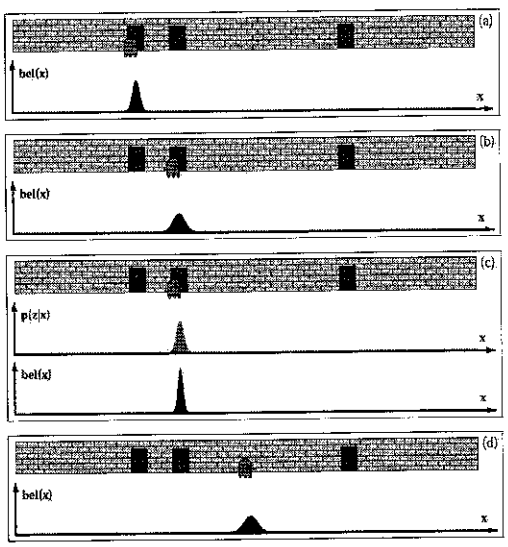



Fig. 4 - Illustration of Kalman filter localization algorithm. In each picture, the belief $bel(x)$ function is represented. In (c) also the observation model $p(z_t|x_t)$ is represented. All densities are represented by uni-modal Gaussians. Image from [4].

As the robot continues to move along the hallway, the uncertainty in its pose increases again, since the filter motion model continues to incorporate the system uncertainty in the belief (Fig. 4.(d)).

When the correspondence variables are unknown, the identity of the landmarks has to be determined during localization. One of the most simple and used strategies is *maximum likelihood* [8] correspondence, in which one first determines the most likely value of the correspondence variable and then applies the filter as described, using the estimated correspondence as granted. With this kind of approximations, the Kalman filter localization can be extended for the global localization problem.

There are a number of works that try to use Kalman filters in various ways, to somehow improve it so it can be adapted for some particular situations. One of them, a combination of Bayesian estimation with Kalman filtering for localization purposes is presented in [9], allowing a better performance of the system by relaxing the Kalman assumptions about the measurements noise.

A work described in [10] applies the Kalman filter basis divided in several smaller communication Kalman filters. Each of the robots has its own filter to process the data from its own sensors. Information exchange between two individual filters is only necessary when two robots detect each other and measure their relative position. In this case, a more comprehensive Kalman filter is used, capable of taking the new information into account.

### C. Monte Carlo localization (MCL)

This is another popular algorithm for localization, which represents the belief $bel(x_t)$ by a set of $M$ particles $X_t = x_t^{[1]}, x_t^{[2]}, ..., x_t^{[M]}$. It is the application of a particle filter to the localization problem. The initial belief $bel(x_0)$ is obtained by randomly generating $M$ pose particles from the prior distribution $p(x_0)$ (usually uniformly over the entire pose space) and assigning each of them a uniform importance factor of $M^{-1}$.

This algorithm in its essence can address both position tracking and global localization problems.

Figure 5 illustrates the MCL in the one-dimensional hallway example. The initial belief is a set of random pose particles uniformly generated for the entire pose space (Fig. 5.(a)). When the sensors are queried and a door is sensed, the MCL algorithm analyses each particle and assigns it an importance factor considering the measurement model, which takes into account the estimated particle, the current observation and the map. In Fig. 5.(b), the resulting particle set is shown, with the height of the particle representing its importance factor. At this stage, the set of particles is identical to Fig. 5.(a), being the importance the only modification. This importance is then used for the re-sampling process, that, based on the set of Fig. 5.(b), generates new particles more concentrated around the most likely positions, but again, all with the same importance factor. Figure 5.(c) shows the new set of particles, also after incorporating a motion.

Again, the new measurement assigns non-uniform importance weights to the particle set, as depicted in Fig. 5.(d). At this point, most of the cumulative probability mass is centered on the second door, which is also the most likely location (and the correct one).

A new motion leads to a new re-sampling phase and a new set is generated according to the motion model (Fig. 5.(e)). Each motion step without measurements tends to disperse the particles, but each motion step merged with measurements tends to concentrate the

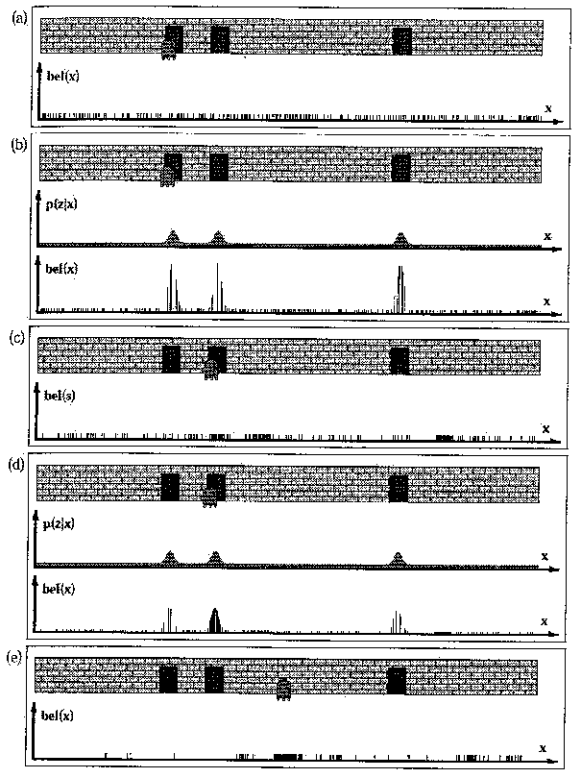



Fig. 5 - Illustration of Monte Carlo localization algorithm. In each picture, the belief $bel(x)$ function is represented. In (b) and (d) also the observation model $p(z_t|x_t)$ is represented. Image from [4].

particles around the correct pose.

MCL has the advantage of not being bound to assumptions about the system and measurement noise (as is the case of Kalman filter, which assumes that the noise is Gaussian). The accuracy of the estimated pose increases with the size of the particle set $M$, although this parameter imposes a trade off between accuracy and computational weight.

A common extension to the MCL algorithm is the addition of an heuristic to add random particles to the particle set. This is done because the particles at places other than the most likely pose gradually disappear. Since at some point they only "survive" near a single pose, there is no way to recover if this pose is incorrect. This kind of approach enables the MCL to solve the kidnapped robot problem.

The use of a Monte Carlo localization algorithm with a variable number of particles is presented in [11], in an attempt to get a good (accuracy / time and computational cost) balance of MCL, for use in a demanding environment like robotic soccer.

Montesano et al. [12] present a study of how vision-based bearings and motion can be used for pairs of robots to localize themselves using each other as landmarks. They try an extended version of Kalman filter, a particle filter approach (MCL) and a combination of both. They observed that the particle filtering approach tends to be more robust than the Kalman filter to estimate the initial location. Once the filters converge to the true location, all methods provide similar results. The combination of both seems to provide the best compromise between robustness and efficiency.

Gutmann and Fox [13] present a comparison between Kalman filtering, Markov localization, Monte Carlo localization and combinations of them in a landmark based scenario, and present some results and comments on several strong and weak points of each approach. In a general way, the combined methods yield better results than the standard versions.

### *D. Tribots localization*

This algorithm was created in the scope of robotic soccer and is based on guided update steps modeling the localization problem as an error minimization task [14]. However, its application can be more general (an example is the work presented in [15]).

It relies on a Look Up Table (LUT) built over the map which, for each position, keeps the minimum distance to the closer landmark. The algorithm starts by assuming a given pose as true and estimates the error between the measured distances (distances of the considered pose to the landmarks represented by the observation model) and the LUT distances (known distances of each position to the defined map landmarks). The objective is to maximize the fitness (minimize the error) of the match between the detected landmarks and the known landmarks (from the map) by gradually correcting the estimation. The pose that will be tested in the next step of the process is given by applying a displacement to the current pose based on the gradient of the current error, using an algorithm named RPROP [16].

Consider the example of its main application, localization of a soccer robot. The landmarks used on the soccer field are the white lines, which are known *a priori*, as they are defined in the rules. The LUT of the map is then based on a representation of the distance of each position to the closer line (Fig. 6).

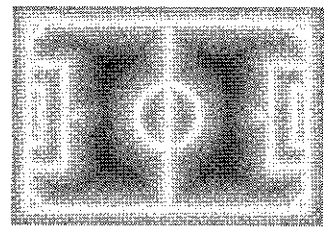
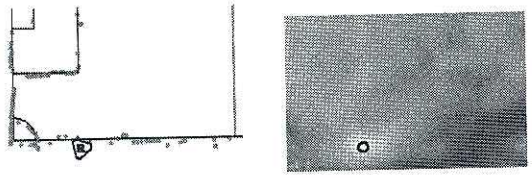


Fig. 6 - Illustration of a LUT map used by the algorithm. Darker areas indicate positions where the distance to the nearest line is large, while brighter areas indicate positions with smaller distances.

In this case, the sensor information is visual. At a given instant, the robot sees a set of points over the field lines. These points (with known distance and position relatively to the robot itself) are the landmarks used in the minimization error function. With a set of line points as in Fig. 7.a), the error function estimates

a position by matching the distance to each line that is measured by the robot with the known LUT distances of that supposed position to the lines. Fig. 7.b) plots the error function as gray levels, being dark areas the zones where the error is larger and brighter areas the zones of smaller error. The most probable position estimate (less error) is indicated by the black circle.



a) b)

Fig. 7 - Left **a)**: The set of line points (gray circles) and the field markings (solid lines) for a localization estimate. The estimated pose for the robot is indicated by the "R" object; Right **b)**: A gray level plot of the error function considering the pose estimation of the left-hand figure. Dark areas indicate positions with large error, bright areas indicate positions with small errors. Image from [14].

Filliat et al. [17] presents a review of localization strategies for map-based navigation, as well as several mapping methodologies.

## IV. SIMULTANEOUS LOCALIZATION AND MAPPING (SLAM)

In plain localization problems, a map of the environment is available for the robot to observe and discover where it is. In SLAM problems, neither the map of the environment or the robot pose are available. In this case, only measurements $z_{1:t}$ and controls $u_{1:t}$. It is a significantly more difficult problem as the robot has to acquire a map of the environment while simultaneously localizing itself within this map.

Mapping is the way the robot builds its representation of the world, its map. Typically, the construction of these maps are based on two methods [18]:

**Metric or grid-based mapping**. It is a representation based on the measures of the space they map. In an indoor metric map, the information included could be the length of wall sections, widths of hallways, distances between intersections, and so on (Fig. 8). With this approach, the robot has to be equipped with sensors capable of estimating the distance to each detected object and it is usual to represent the mapped space with an evenly spaced grid. This is a quantitative approach and a path plan that could be applied over this kind of map could be "advance for X meters, turn $\theta$ degrees and advance other Y meters";

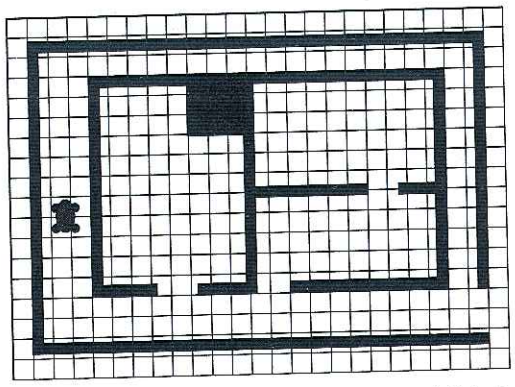


Fig. 8 - Illustration of a metric or grid-based map. All the known environment objects have known measures and positions.

**Topological mapping**. The representation of the space is not based on precise measurements but rather on landmarks (Fig. 9). In an indoor topological map, the information could include doors, hallway intersections, or T-junctions of hallways. To build a map based on landmarks, the robot would need to be equipped with sensors that could identify them, typically vision systems. This is the most intuitive approach for humans, as we typically use path plans like "go straight till you see *this landmark*, turn left after it and follow until you reach *that landmark*";

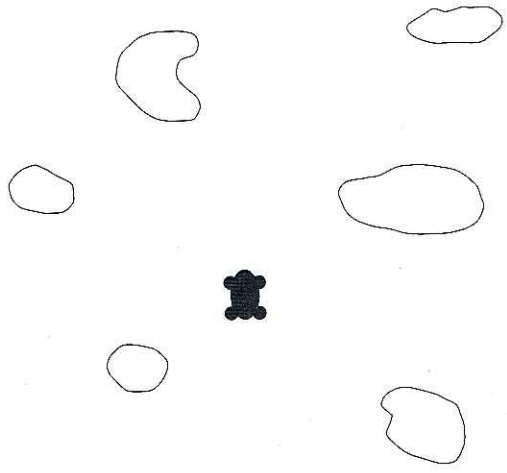


Fig. 9 - Illustration of a landmark map. The known landmarks are positioned relatively to each other, but no precise measures are known, navigation is only possible by sighting of the landmarks.

From a probabilistic perspective, the SLAM problem has two main forms.

One is known as *online SLAM*, which consists of estimating the posterior probability over the current pose along with the map: $p(x_t, m|z_{1:t}, u_{1:t})$. It involves the estimation of the variables that persist at time $t$. It is common that online SLAM algorithms discard past measurements and controls once they have been processed.

The other main form is *full SLAM*, which consists of estimating the posterior over the entire path $x_{1:t}$ along with the map, instead of just the current pose $x_t$: $p(x_{1:t}, m|z_{1:t}, u_{1:t})$.

Historically, the earliest SLAM algorithm is based on the extended Kalman filter (EKF) [4]. The EKF SLAM algorithm applies the EKF for online SLAM by using maximum likelihood data association. It is an approach that imposes a number of limiting assumptions: (1) the map has to be composed of point landmarks, preferably with as little ambiguity as possible, which may require significant attention to feature detectors; (2) as other Kalman algorithms, the assumption of Gaussian noise for both the robot motion and perception models needs to be met (or acceptable); (3) the algorithm can only process positive sightings of landmarks, no processing can be made based on the absence of landmarks.

Another example of approach for SLAM presented in [4] is the Extended Information Form (EIF) algorithm. This EIF SLAM algorithm has in common with EKF SLAM the fact that it represents the posterior estimation by a Gaussian. However, unlike EKF SLAM, EIF SLAM solves the full SLAM problem. Thus, instead of defining the posteriors over the map and the most recent pose, it defines the posteriors over the map and the entire robot path. The EKF SLAM is therefore an incremental approach, it enables the robot to update its map without memory concerns, while the EIF SLAM is best suited for problems where we want a map from a fixed set size and we can afford to hold the data in memory up to the time where the map is built.

Several other algorithms are currently available, like fastSLAM [19], Postponement [20], relative map representations 16, submap methods 17, Schmidt-Kalman filter based 19, Covariance Intersection (CI) based methods 1O, and decorrelation methods 11

In [21] a distributed extended version of Kalman filter is used for cooperative multi-robot localization and mapping in outdoor environments, providing better estimates for each robot localization based on relative information of other robots poses as well as increased mapping capabilities since by sharing the own information, each robot can get unknown pieces of the map to fit into its own map and, in case of overlapping pieces, merge the information to get better estimates.

Other algorithms and approaches for multi-robot localization and mapping are presented in [22], [23] and [24]. An application of multi-robot localization but applied to object localization, rather than self localization is presented in [25].

In [26], the authors propose a hybrid approach on SLAM, with several filters implementing their own SLAM algorithm and with the ability of exchanging information between them, before generating the output.

## V. FINAL REMARKS

The localization problem for mobile robots is a critical feature that is generally not trivial to solve. Furthermore, most mobile robotics applications have specific functions to fulfill, meaning that the localization is one of the needed features but not the objective, and thus have to be accomplished effectively independent of the conditions of the robot surroundings and movement.

Several kinds of approaches exist for addressing this problem, all of them with their own drawbacks. Almost every applications need to make compromises when choosing a localization technique.

When there is not the possibility to provide a map to the robot, the problem escalates to a Simultaneous Localization And Mapping, which is an even more challenging issue.

In this paper, some key aspects of these problems have been presented, as well as brief descriptions of some of the most used techniques.

## REFERENCES


[1] N. Metropolis and S. Ulam, "The Monte Carlo method", *Journal of American Statistical Association*, vol. 44, no. 247, pp. 335–341, 1949.

[2] R.E. Kalman, "A New Approach to Linear Filtering and Prediction Problems", *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[3] E.A. Wan and R. Van Der Merwe, "The unscented Kalman filter for nonlinear estimation", in *IEEE Adaptive Systems for Signal Processing, Communications, and Control Symposium*, 2000, pp. 153–158.

[4] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*, The MIT Press, 2005.

[5] B. Siciliano and O. Khatib, Eds., *Springer Handbook of Robotics*, Springer, 2008.

[6] D. Fox, W. Burgard, and S. Thrun, "Markov localization for mobile robots in dynamic environments", *Journal of Artificial Intelligence Research*, vol. 11, pp. 391–427, 1999.

[7] G. Bishop and G. Welch, "An Introduction to the Kalman Filter", in *Proc of SIGGRAPH, Course 8*, Chapel Hill, NC, USA, 2001, number NC 27599-3175.

[8] A. Howard, M.J. Mataric, and G.S. Sukhatme, "Localization for mobile robot teams using maximum likelihood estimation", in *IEEE International Conference on Intelligent Robots and Systems*, 2002, pp. 434–459.

[9] S.I. Roumeliotis and G.A. Bekey, "Bayesian estimation and Kalman filtering: A unified framework for mobile robot localization", in *IEEE International Conference on Robotics and Automation*, 2000, vol. 3, pp. 2985–2992.

[10] S.I. Roumeliotis and G.A. Bekey, "Distributed multirobot localization", *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 781–795, 2002.

[11] P. Heinemann, J. Haase, and A. Zell, "A combined monte-carlo localization and tracking algorithm for robocup", in *IEEE International conference on Intelligent Robots and Systems*, October 2006, pp. 1535–1540.

[12] L. Montesano, J. Gaspar, J. Santos-Victor, and L. Montano, "Cooperative localization by fusing vision-based bearing measurements and motion", in *IEEE International Conference on Intelligent Robots and Systems*, August 2005, pp. 2333–2338.

[13] J.-S. Gutmann and D. Fox, "An experimental comparison of localization methods continued", in *IEEE International conference on Intelligent Robots and Systems*, 2002, vol. 1, pp. 454–459.

[14] M. Lauer, S. Lange, and M. Riedmiller, "Calculating the perfect match: an efficient and accurate approach for robot self-localization", in *RoboCup 2005: Robot Soccer World Cup IX*, Ansgar Bredenfeld, Adam Jacoff, Itsuki Noda, and Yasutake Takahashi, Eds., vol. 4020 of *Lecture Notes in Computer Science*, pp. 142–153. Springer, 2006.

[15] Manuel Gouveia, António Paulo Moreira, Paulo Costa, Luís Paulo Reis, and Marcos Ferreira, "Robustness and precision analysis in map-matching based mobile robot self-localization", in *New Trends in Artificial Intelligence: 14th Portuguese Conference on Artificial Intelligence*, 2009, pp. 243–253.

[16] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: the RPROP algorithm", in *IEEE International Conference on Neural Networks*, 1993, vol. 1, pp. 586–591.

[17] David Filliat and Jean arcady Meyer, "Map-based navigation in mobile robots: I. A review of localization strategies", *Cognitive Systems Research*, vol. 4, no. 4, pp. 243–282, December 2003.

[18] G.A. Bekey, *Autonomous Robots: From Biological Inspiration to Implementation and Control*, The MIT Press, 2005.

[19] Michael Montemerlo, Sebastian Thrun, Daphne Koller, and Ben Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem", in *AAAI National Conference on Artificial Intelligence*. 2002, pp. 593–598, AAAI.

[20] A.J. Davison, *Mobile robot navigation using active vision*, PhD thesis, University of Oxford, 1998.

[21] R. Madhavan, K. Fregene, and L.E. Parker, "Distributed cooperative outdoor multirobot localization and mapping", *Autonomous Robots*, vol. 17, no. 1, pp. 23–39, July 2004.

[22] Ioannis M. Rekleitis, Gregory Dudek, and Evangelos E. Milios, "Multi-robot cooperative localization: A study of trade-offs between efficiency and accuracy", in *IEEE International Conference on Intelligent Robots and Systems*, 2002, pp. 2690–2695.

[23] A.I. Mourikis and S.I. Roumeliotis, "Performance analysis of multirobot cooperative localization", *IEEE Transaction on Robotics*, vol. 22, no. 4, pp. 666–681, August 2006.

[24] Sebastian Thrun, "A probabilistic online mapping algorithm for teams of mobile robots", *International Journal of Robotics Research*, vol. 20, no. 5, pp. 335–378, 2001.

[25] Ashley W. Stroupe, Martin C. Martin, and Tucker Balch, "Merging gaussian distributions for object localization in multi-robot systems", in *Seventh International Symposium on Experimental Robotics*. 2000, Springer-Verlag.

[26] S.J. Julier and J.K. Uhlmann, "Using multiple SLAM algorithms", in *IEEE International Conference on Intelligent Robots and Systems*, October 2003, vol. 1, pp. 200–205.