Spatial Navigation Algorithms for Autonomous Robotics

Advanced
Seminar

submitted by
Chiraz Nafouki

NEUROSCIENTIFIC SYSTEM THEORY
Technische Universität München

Supervisor: Ph.D. Marcello Mulas

Final Submission: 07.07.2015

Advanced Seminar:
Spatial Navigation Algorithms for Autonomous Robotics

Problem description:

Recent advancements in hardware technology opened up new possibilities for the use of robotic vehicles that were unimaginable until a few years ago. However, an obstacle to the spread of autonomous robots that can freely explore complex environments is the need of considerable computational resources to localize the robots in space and to build at the same time a map of the environment (Simultaneous Localization And Mapping – SLAM). Current existing algorithms are still far from matching performance achieved by humans.

Task:

With this advanced seminar two students shall review the main techniques used by existing SLAM algorithms. More precisely, one of the students should focus on engineered systems that are based on traditional techniques. The other student should focus on algorithms that take inspiration from biology in order to improve robotic spatial navigation.
Each student should illustrate the capabilities and limitations of each class of SLAM with particular reference to scalability, estimation accuracy and computational performance.
In conclusion the students should jointly compare strengths and weaknesses of bio-inspired SLAMs in reference with more standard approaches.
In addition, the students should provide an overview of validation methods used to compare the performance of different SLAM algorithms.

In summary the students shall:

- give an overview of the main classes of SLAM algorithms
- compare standard engineered systems with bio-inspired strategies
- illustrate validation methods used to compare different SLAM algorithms

Supervisor: Marcello Mulas

Abstract

This report presents the two main categories of Simultaneous Localization and Mapping (SLAM) algorithms : probabilistic algorithms and bio-inspired algorithms. It presents some characteristics, advantages and disadvantage of each category, compares them based on a number of validation methods and gives examples of applications for each method.

Contents

Chapter 1

Introduction

The problem of Simultaneous Localization and Mapping, or SLAM, has attracted an immense attention in the robotics community. SLAM addresses the problem of a mobile robot moving through an unknown environment. The robot should build a map of the environment while simultaneously determining its location within the map. These two simultaneous tasks should usually be realized online for an autonomous real-time navigation.

In order to achieve this, the robot can use both external cues (also known as allothetic sources) and internal sources (also known as idiothetic sources). External cues usually mean observations of features in the environment. These observations are taken by the sensors of the robot like a camera, a microphone, laser, lidar or sonar. The choice of the type of sensors depends on the type of environment and the SLAM algorithm used. The main problem with all these sensors is that they always carry noise which results in measurement errors. Internal sources represent the kinematic model and odometry information that make the robot aware of its own motion, such as wheel rotation measurements. Neither of idiothetic or allothetic sources can usually be used alone. For example, in a building, it is nearly impossible to determine a location solely with the visual information, because all the corridors may look the same. It is also almost impossible for the robot to achieve SLAM based solely on external cues without knowing its motion.

There is a very broad range of applications of SLAM algorithms. Both indoor and outdoor applications for both manned and autonomous vehicles can be found in the SLAM literature. However SLAM is mostly used to ensure the navigation of autonomous robots. The applications vary from a simple autonomous vaccum cleaner to underwater drones or space vehicles.

Despite the important improvement achieved in the understanding of the SLAM problem in the recent years, it is still considered as one of the biggest challenges in robotics and autonomous navigation. The difficulty of SLAM is due to the fact that the map and robot pose estimates are generally correlated but in SLAM both map and position are initially unknown. This is known by the name of the chicken-egg problem, which means that a map is needed for localization and a pose estimate is needed for mapping.
One other challenge for the SLAM problem is data association. This means that two different features or landmarks in the environment can be perceived as the same by the robot or that a previously seen landmark by the robot is considered as a new landmark. The data association issue is currently solved using some feature extraction and matching algorithms such as SIFT and SURF, which seem to be considerably good solutions. However they become computationally expensive when dealing with high dimensional maps due to the high cost of extracting and matching features.

Despite these challenges, more efficient, consistent and robust solutions have been provided over the last two decades to solve the joint localization mapping issue. The most known solutions are classified into two categories. One category consists of a number of mathematical probabilistic techniques, where the representation of the observations and the motion models is generally performed by computing its prior and posterior distributions . The other category focuses on emulating the biological systems thought to be responsible for mapping and navigation in animal and human brain.

In this report, we present these two main classes of algorithms. Regular or probabilistic algorithms are presented in chapter 2 and bio-inspired algorithms are presented in chapter 3. We also give examples of applications of these two categories within each of the chapters. In chapter 4, we try to make a comparison between the two SLAM categories based on some validation methods that we also present in chapter 4. Finally we conclude the report in chapter 5 and present some possible future improvements for SLAM.

Chapter 2

Overview of standard SLAM algorithms

As we mentioned previously, the motion of the robot increases the uncertainty of the position and map estimations and adds errors and noise to the observations. Therefore, probabilistic approaches are usually used in SLAM problems to model and reduce the uncertainty and noise in order to make decisions concerning the estimation of the map and location of the robot.

The most used SLAM algorithms nowadays are probabilistic algorithms. In fact, since the 1990s probabilistic approaches have become dominant in SLAM because they allowed the robotic community to reach levels of autonomy and robustness that had not been reached before. As its name suggests, a probabilistic algorithm is based on the idea of representing information through probability densities.

Almost all the probabilistic techniques are based on the Bayes rule: $p(x/d)\,p(d)=p(d/x)\,p(x)$. This formula tells us that if want to learn about a quantity x (for example a robot position or a map) based on measurement data d (it can be odometry, visual data...), we can do that by multiplying the two terms $p(d/x)$ and $p(x)$. The term $p(d/x)$ describes the way a sensor measurement d of a real-world value x is generated. In other words, it is the probability of observing the measurement d under the hypothesis x . $p(x)$ is called the prior. It specifies the probability that x is the value in the real world before the arrival of any data. $p(d)$ can be viewed as a normalization constant.

For a typical SLAM situation, data arrives over time. Thus, the robot uses Bayes filters recursively to estimate its position and the environment state. If we want to express the aim of the robot mathematically, we can say that it is to calculate the probability distribution: $p(x_{(0:T)}, m/z_{(1:T)}, u_{(1:T)})$,where x_k is the robot position at time k (k varying between 0 and T, where x_0 corresponds to the initial position of the robot and x_T to the last position), m is the map of the environment (supposed to be static in time), z_k is an observation measured by robot sensors at time k and u_k is a motion command given to the robot at time k (for example go one meter straight or turn right...). x_k is generally represented by three coordinates in case of a two dimensional map: the Cartesian coordinates in the plane x and y , and the heading direction of the robot θ .

In the probability distribution mentioned above, $x_{(0:T)}$ describes the successive positions i.e the path followed by the robot over discrete time by the robot. The map m is a collection of landmarks $m_{(1:N)}$, with N being the number of landmarks.

Each landmark is represented by its Cartesian coordinates in the map. For example in case of a two dimensional map, $m=(m_{(1,x)}, m_{(1,y)}, m_{(2,x)}, m_{(2,y)} \ldots, m_{(N,x)}, m_{(N,y)})$.
During the motion of the robot, the pose probability distribution $p(x_{(0:T)})$ is updated using odometry data and the robot's motion model. Therefore the robot needs to know well its kinematic model and the sensors need to measure accurately the motion of the robot in order to avoid the accumulation of errors. Observations of landmarks are used to update not only the landmark location distributions, but also the robot pose distribution.

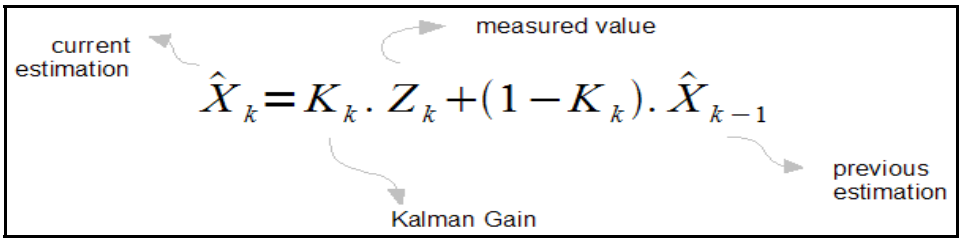
1. Main techniques used by regular SLAM algorithms

There are a big number of well-known SLAM approaches that use probabilistic filters. The most known approaches are based on Kalman Filter (KF) and its variations, Particle Filters (PF) and its variations and Expectation Maximization (EM) algorithms.
In this section, we present these filters and algorithms that were proven to be successful at reducing the amount of uncertainty and noise inherent to the robot motion and to the sensors measurements. We also present some advantages and limitations of these standard SLAM strategies and give some examples of their applications.

1.1. Kalman filter and its extensions

1.1.1 Kalman filter

Kalman filter (KF) is a kind of Bayes filter which represents posteriors at time k $p(x_k, m/z_k, u_k)$ using Gaussians. Gaussians are unimodal distributions that can be represented compactly by a small number of parameters. In our case, the Gaussian model is the full state vector X_k , which c the robot's pose x_k and the map: $X_k^T=(x_k, m)$.
In order to estimate the state vector X_k , Kalman filter is based on the following estimation:



where $X_k^{\wedge}$ denotes the current estimated state vector at time k , $X_{(k-1)}^{\wedge}$ is the previous estimated state vector at time $k-1$, Z_k is the measured observation and K_k is a constant representing the gain of the filter. As we can notice in this equation, KF is based on the assumption that the state transition (from k to $k-1$) is a linear function. For each new update of the state vector, the unknown gain K_k needs to be determined in the most accurate way in order to get a correct new estimated state.
Besides the general previous estimation, the two state equations used in KF model are :

$$X_k = AX_k - 1 + Bu_k + w_{(k-1)} \quad (1)$$

$$z_k = HX_k + v_k \quad (2)$$

Here A , B and H are constant matrices. $w_{(k-1)}$ and v_k are two added Gaussian noises statistically independent representing the noise of system and observation respectively.
The basic idea in these two equations is that the robot uses the known states to predict the states of the next step, and then this prediction is corrected based on the observation at the next step.

The first equation tells us that X_k is a linear combination of its previous value $X_{(k-1)}$, the motion control signal u_k and the noise $w_{(k-1)}$ related to the motion of the robot.
The second equation tells us that the any measurement value z_k is a linear combination of the state X_k and the measurement noise v_k which expresses the noise in sensors.
KF is composed of two steps at each iteration in time: the time update or prediction step and the measurement update or correction step (Fig.1)

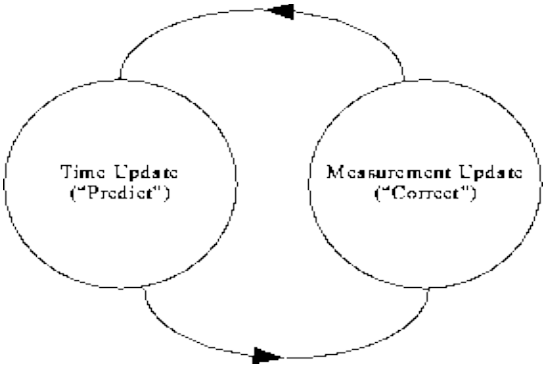


Fig.1.The two steps in Kalman filter: prediction and correction

In the prediction step, the prior estimate state vector $X_k^{\wedge\prime}$ i.e a rough prediction of the estimated state vector $X_k^{\wedge}$ is calculated based on equation(1). The "prior error covariance" matrix is also predicted during this step. These priors values are used by the correction step that follows. During the measurement update step, the Kalman gain is computed using equation(2) and the estimated state vector $X_k^{\wedge}$ is calculated using the updated gain value. Finally the covariance error matrix is also updated and the two corrected values at the end of the correction step (also called posterior values) will be used in the prediction step of the next iteration. The two initial posteriors of the KF (i.e in the first iteration) are also supposed to be Gaussians.

KF performs well in case the variables are linear the noise is Gaussian but it has a high computational cost especially in high dimensional map. In fact, Kalman filter-based algorithms require time quadratic in the number of landmarks in order to incorporate each sensor observation.
There are many SLAM probabilistic algorithms that use the KF, mainly for the map estimation in a static environment (the landmarks positions do not change with time). This is for example the case for an algorithm called vSLAM which relies on both odometry and visual data to perform SLAM.

There are many variations of the KF in the state-of-the-art SLAM. The main variations of KF are the Extended Kalman Filter (EKF) , Information Filtering (IF) and

Extended IF (EIF). In the next sections we present some of the most used KF variations and some of their advantages and disadvantages.

1.1.2 Extended Kalman filter

As we have seen for the KF, the current position of the robot x_k is supposed to depend linearly on the previous position value $x_{(k-1)}$ and the control u_k . However, in reality, there is usually a non-linear dependence between these variables. In order to deal with this limitation, the KF was extended to a filter called the Extended Kalman Filter (EKF), which is used by many SLAM approaches because it is able to handle with non-linear models. To accommodate such non-linearities, Kalman filters approximate the robot motion model using a linear function obtained via Taylor series expansion. These approximations works generally well but sometimes it can introduce large errors in the estimated posterior and covariance matrix, which may lead to divergence of the filter.

Besides, like the KF, EKF has a high complexity (time quadratic in function of the number of landmarks). There have been some improved versions of the EKF such as the one used in the Devide and Conquer SLAM algorithm (D&C SLAM), which reduces the total computational cost of updating the EKF from $O(n^2)$ to $O(n)$ while maintaining precision. This algorithm was even shown to have better consistency properties than standard EKF SLAM since the state covariance computed by D&C SLAM represents more accurately the real estimation error.
One of the most famous algorithms that use EKF is FastSLAM which uses a combination of particle filtering for the path estimation and EKF for the map estimation inorder to reduce the computational cost and achieve a robust and accurate estimation at the same time. Both Particle Filter and FastSLAM will be discussed in the next sections.

1.1.3 Information filter and Extended information filter

From an analytical point of view, the Information filter (IF) is equal to the Kalman filter. The difference is that IF keeps track of the inverse of the covariance matrix called information matrix rather than the common covariance matrix. The advantage for using this matrix is that inverse covariance matrix is often sparser than the covariance matrix and therefore faster to compute. IF was also proven to be more stable and accurate than KF. However, for high dimensional spaces the IF was shown to be computationally more expensive than KF. This is one of the reasons why the KF has been vastly more popular than the IF.
The Extended information filter (EIF) is, like the EKF, capable of handling non-linear situations. Like the IF, instead of maintaining the covariance matrix, EIF maintains the inverse covariance matrix or information matrix.
IF and EIF are used in many SLAM applications especially in multi-robot SLAM applications. For example, EIF was used by Gerasimos G. Rigatos in the UK for coordinating autonomous agricultural robots which collaborated under a master-slave scheme to achieve corn harvesting.

1.1.4 Unscented Kalman filter

The Unscented Kalman Filter (UKF) addresses the approximation issues of the EKF and the linearity assumptions of the KF. The basic difference between the EKF and UKF stems from the manner in which Gaussian random variables (GRVs) are represented for propagating through system dynamics. In the EKF, the initial state distribution of the

robot is approximated by a GRV, which is then propagated analytically through the linearization of the nonlinear system. This can introduce large errors in the calculation of posterior mean and covariance of the transformed GRV, which may lead to sub-optimal performance and sometimes divergence of the filter. In the UKF, the state distribution is like in EKF represented by a GRV, but is specified using a minimal set of carefully chosen sample points. These sample points completely capture the true mean and covariance of the GRV, and when propagated through the non-linear system, captures the posterior mean and covariance accurately. Thanks to this deterministic sampling approach, UKF's performance is superior to that of EKF which provides only a first-order approximation to optimal nonlinear estimation. Remarkably, the computational complexity of the UKF is the same order as that of the EKF.
UKF was used instead of the EKF for state estimating by Niko Sunderhauf et al. in a SLAM application at a university campus at Chemnitz, Germany with an autonomous airship called “Fritz” that uses a monocular camera (Fig.2).

Fig.2. Airship "Fritz" used to test UKF during a flight above a university campus at Chemnitz, Germany

1.1.5 Compressed extended Kalman filter

The Compressed extended Kalman filter (CEKF) gives an estimation that is identical to the standard EKF but dramatically reducing the computational cost and memory requirement from $O(n^2)$ to $O(n)$. CEKF works by computing local KFs (on local maps) and then updating its output to a global map. By this way, it only needs to handle with small amounts of data during the local iteration process. Therefore, for large scenarios, or big maps with high population of landmarks, the CEKF seems to perform better than KF and its other variations.
This algorithm was tested by Jose Guivant and Eduardo Nebot with a vehicle in a large outdoor unstructured environment (Fig.3) and showed a remarkable improvement when it comes to computational cost.

Fig.3. Utility vehicle used to test UKF in a large outdoor environment

1.2. Particle filter and its extensions

One advantage of KF and EKF that we presented in previous sections is that they are very accurate: they usually provide optimal Minimum mean-square Error (MMSE) estimates of the robot state (i.e robot and landmark positions). However, the Gaussian noise assumption restricts significantly the adaptability of the KF and EKF because in real world the noise is not always Gaussian. The particle filter (PF), also called the sequential Monte-Carlo (SMC) method, offers a solution to this important limitation since it enables to approximate the posterior density function of the state of the robot in case of non linear and non-Gaussian input. In the following, we present the PF and its most known extension : the Rao-Blackwellized particle filter.

1.2.1 Particle filter

Particle Filter (PF) is a recursive Bayesian filter which uses Monte-Carlo method. In contrast to parametric filters like KF, PF represents the state distribution and the posterior $p(x_{(1:T)}, m / z_{(1:T)}, u_{(1:T-1)})$ by a set of random point clusters called 'particles' , where each particle represents a potential trajectory of the robot. The map is then built from the observations and the trajectories represented by the particles. PF is capable of handling highly non-linear sensors and non-Gaussian noise. The PF is also easy to implement and tune. However, this ability to treat non-linearity and non-Gaussian noise produces a big growth in computational complexity with the state dimension as new landmarks are detected, which makes PF not suitable for real time applications. In fact, the number of particles needed to represent a posterior grows exponentially with the dimension of the state space which leads to a very high computational cost. For this reason, PF has only been successfully applied to localization, i.e. determining position and orientation of the robot, but not to map-building, i.e. landmark position and orientation. Only limited works use PF for the whole SLAM algorithm. Most of works that use PF use it in combination with other techniques, mainly KF or EKF. This was the case for the well-known FastSLAM algorithm and its improved versions.

1.2.2 Rao-Blackwellized particle filter

The Rao-Blackwellized (marginalized) particle filter (RBPF) makes use of the following factorization: $p(x_{(1:T)}, m/z_{(1:T)}, u_{(1:T-1)}) = p(m/x_{(1:T)}, z_{(1:T)})\, p(x_{(1:T)}/z_{(1:T)}, u_{(1:T-1)})$.

This factorization makes it possible to first estimate only the trajectory of the robot $p(x_{(1:T)}/z_{(1:T)}, u_{(1:T-1)})$ and then to compute the map given that trajectory $p(m/x_{(1:T)}, z_{(1:T)})$. Since the map strongly depends on the pose estimate of the robot, this approach offers an efficient computation and can therefore handle high-dimensional problems compared with PF.
RBPF is used in many SLAM applications up to date. For example, it was successfully applied to the high-dimensional navigation system of the Swedish fighter aircraft Gripen. Very good results on a real flight were obtained with 5000 particles for the marginalized filter and described by Thomas Schön et al.

1.3. Expectation Maximization algorithms

Expectation Maximization (EM) estimation is a statistical algorithm that was developed in the context of maximum likelihood estimation. It iterates two steps: an expectation step (E-step), where the posterior over robot poses is calculated for a given map, and maximization step (M-step), in which the most likely map is calculated given these pose expectations. The final result is a series of increasingly accurate maps.
EM offers an optimal solution for map-building, but not for localization. In fact, it is able to build a map when the robot's pose is known. The main advantage of EM with respect to KF is that it can tackle the data association problem very well. This is possible thanks to the fact that it localizes repeatedly the robot relatively to the map in the E-step, generating various hypotheses concerning the possible position of the robot. In the M-step, these hypotheses are translated into features in the map, which then get reinforced in the next E-step or gradually disappear. However, the need in EM to process the same data several times to obtain the most likely map makes it inefficient, not incremental and not suitable for real-time applications. Even by using discrete approximations to estimate the robot's pose, the cost of EM grows exponentially with the size of the map. Besides, the error increases significantly with size and the resulting map becomes unstable after long cycles. These problems can however be avoided if the data association is known. Some practical applications use EM to construct the map (only the M-step), while the localization is done by different means, for instance using PF to estimate the pose of the robot from odometer data.
EM was used by Sebastian Thrun in an application to generate three-dimensional maps of indoor environments based on camera measurements acquired by a mobile robot. It was also successfully used by Wolfram Burgard et al. in a sonar-based SLAM application which allowed them to construct large maps of some indoor environments using sonars which are considered to be inaccurate sensors.

2. Example of traditional SLAM algorithms: FastSLAM

FastSLAM is an algorithm based on Rao-Blackwellized particle filtering. Like RBPF, FastSLAM decomposes the SLAM problem into a robot localization problem, and a collection of landmark estimation problems that are conditioned on the robot pose estimate. This factored representation is exact because landmark estimates become conditionally independent given the robot's path.

FastSLAM uses particle filter for estimating the posterior over robot paths. Each particle of the PF makes its own local data association and possesses one or more Kalman filters that estimate the landmark locations conditioned on the path estimate.
FastSLAM scales logarithmically with the number of landmarks in the map. In fact, the use of PF for the path estimate requires less memory usage and computational time than a standard EKF or KF. Therefore, FastSLAM algorithm was run successfully on 50,000 landmarks, a result which was not achieved by the previous standard approaches. Some improved versions of FastSLAM like FastSLAM2.0 can handle even larger environments than the standard FastSLAM.

Chapter 3

Overview of bio-inspired algorithms

Although most of the current work in visual SLAM uses probabilistic techniques, the recent bio-inspired techniques were proven to be able to solve the SLAM problem as well, using methods that are apparently robust, flexible, and well integrated into the robot.
The bio-inspired algorithms, as their name suggests, are inspired by biological systems, mainly the human and rodent's brain. In fact, mammals and insects show the ability to store and organize visual cues, so that a single visual cue, or a brief sequence of cues, can update the pose estimate and relocalize the animal. Rodents, in particular, are better than human in facing navigation problems: rats can navigate and update their representation of pose even without external cues, only by using estimates of self-motion or what is known as path integration. They also have a remarkable ability to store and recall visual cues.
In a parallel manner, bio-inspired SLAM systems use both odometry (idiothetic sources) and observations obtained by vision systems such as lasers or cameras (allothetic sources) to update and correct the estimates of the robot pose and the map. The main difference between these systems and rats is that unlike robots, rats do not build a detailed geometrical representations of the environment.

Most of the studies on rodents have been focused on the particular region of the brain in and around the hippocampus. While rodents also rely on olfaction and whisking, most of the experiments are designed to remove these cues and to focus on the rodent's behavior in a controlled environment. Observations of rodent behavior are accompanied by neural recordings in single or small groups of cells as a rodent moved around in a small arena. These studies have led to the three important discoveries of cells which are responsible for the mapping and localization tasks in the brain: place cells, head direction cells, and grid cells.
In this chapter we present, the basic biological concepts and models that were used by bio-inspired algorithms and give two examples of successful bio-inspired SLAM algorithms which are RatSLAM and a bio-inspired neurol model of extended Kalman filter.

1. Main techniques and models used by bio-inspired SLAM algorithms

1.1. Place cells

In the early 1970s, place cells were discovered by John O'Keefe of University College, London. They are located in the hippocampus (Fig.1). These cells fire consistently when the animal or the human is at a particular location in the environment, but fire much less as the rodent moves away from this location. In fact, the firing of each place cell signals recognition of a specific place in a familiar environment. This specific place is known as the cell's 'firing field'.

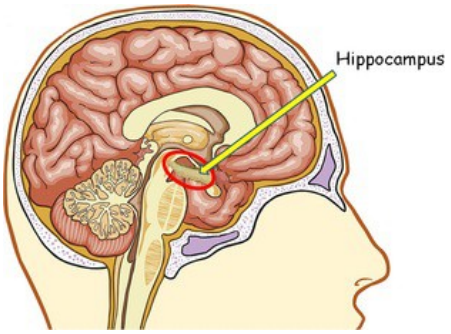


Fig.4. location of the hippocampus in the brain

John O'Keefe and Lynn Nadel suggested that thousands of place cells, covering the surface of any space, acted as a mapping system, and suggested that the hippocampus as a whole created a "cognitive map". This was confirmed by the fact that in case of hippocampal damage, rats lose their ability to solve the spatial problem and get lost in space. Each place cell receives two different inputs, one external input about a large number of environmental stimuli and external events, and an internal input from a navigational system which calculates where an animal is in an environment independently of the external cues using its self-motion.

1.2. Grid cells

Grid Cells were discovered roughly 30 years later after the place cells in the laboratory of Edvard and May-Britt Moser in Norway. Grid cells are found in the medial entorhinal cortex (MEC), a region on the posterior wall of the rat's cerebral hemisphere which is closely related to the hippocampus. (Fig.2)

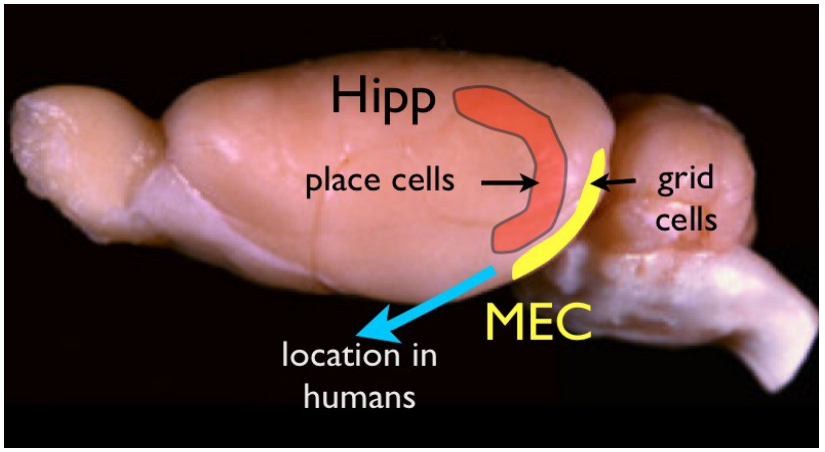


Fig.5. Lateral view of the rat brain. MEC=medial entorhinal cortex. Hipp=hippocampus

Grid cells show place cell-like properties but with multiple firing fields : they fire in a metrically regular way across the whole surface of a given environment. The firing fields of these cells looks very much like hexagons (Fig.3). In fact, a single grid cell will fire when the rat is located at any of the vertices of a tessellating hexagonal pattern across the environment. Grid cell firing appears to be a signal that can be used for measuring displacement distances and direction. In other words, a 'metric'.

There are also conjunctive grid cells which fire only when the rat is at certain locations and in a specific orientation.

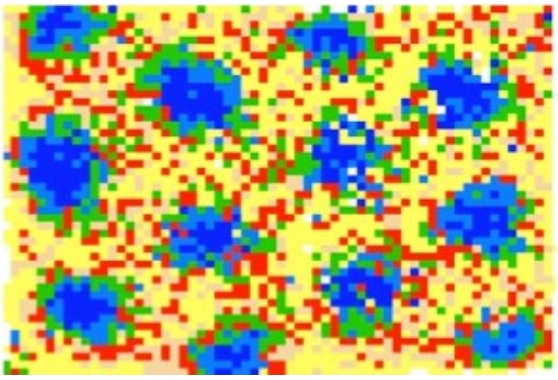


Fig.6. Firing rate maps of a grid cell. Red ans blue pixels represent high average firing rate; yellow pixels are 0 spike/sec firing rates.

1.3. Head direction cells

In the early 1990s, head direction cells were discovered in the neighboring regions of the hippocampus. These cells respond to the head direction of the animal: they fire when the rodent's head is at specific global orientations, but their firing is not correlated to the location of the animal's body. The place cells and head direction cells can be thought of as complementary cell types, one providing positional information, the other directional.

1.4. Continuous attractor networks

Continuous attractor networks (CANs) are often used to model the behavior of place, head direction, and grid cells, especially in robotics. A CAN is a type of neural network that has an array of neural units with weighted connections. The neural units compute activity by summing the activity from other neural units through the weighted connections. The recurrent connections in CANs cause the network to converge over time to certain states (attractors) in the absence of external input.

1.5. Path integration

Path integration is the process used by animals to calculate their current position by using a previously determined position and the estimated speed. Studies reveal the existence of highly effective path integration mechanisms in animals that depend on the directional heading (using head direction cells) and distance computations(using grid and place cells).

In SLAM algorithms, errors in path integration accumulate over time with the motion of the robot due to noise and uncertainties. However, unlike probabilistic SLAM, path integration with a CAN does not carry a representation of the uncertainty accumulated over time. The width and height of the activity packet in a CAN stays constant under path integration.

2. Examples of bio-inspired SLAM algorithms

2.1. RatSLAM

RatSLAM, is one of the most known bio-inspired SLAM algorithms which offer a solution to the problem of large-scale mapping and localization for a vision-only systems. It uses a simplified computational model of the rodent hippocampus to build an online map in real-time. RatSLAM corrects cumulative errors in odometry by a map correction algorithm. Visual ambiguity and data association problems are managed by maintaining multiple competing robot pose estimates.

The pose cells are the biologically inspired part in RatSLAM and mimics the behavior of place cells and head direction cells. Activity in the pose cells is updated by self-motion cues, and calibrated by local view. The self-motion cues are used to drive path integration in the pose cells, while external cues trigger local view cells that are associated with pose cells.
The core of the system is a 3 dimensional continuous attractor network, called the pose cell network (PCN). Due to the attractor dynamics, self-preserving packets of local activity form in the network of pose cells. These local packets compete, trying to annihilate one another until a stable state is reached. The pose cell network is used to maintain an estimate of the system's current pose.
RatSLAM was able to map a complex suburban road network of 66 km using a single webcam operating at 10 Hz mounted on a car. RatSLAM generated a coherent map of the entire environment at real-time speed and closed correctly more than 51 loops of up to 5 km in length. This result hasn't been achieved by any of the probabilistic SLAM algorithms.

2.2. A bio-inspired neural model for Extended Kalman filter

One disadvantage of the Extended Kalman filter is that its robustness decreases when the noise model is inaccurate or unknown. In fact, unknown external noise may result in the inaccuracy of the state estimate, and even cause the divergence of EKF. To solve this problem, a bio-inspired neural model for EKF was proposed in 2014 by Jianjun Ni et al. This model was integrated in EKF algorithm and used to adjust the system noise and observation noise (see section 1.1.1.) adaptively inorder to reduce the estimation error of EKF. The system noise and observation noise are changed at each iteration of the EKF depending on the difference between the prediction of the system state and the real value obtained by the robotic sensors.
The algorithm was proven to work well without any prior knowledge of the noise model, nor any learning procedures. Experimental results showed the efficiency of this approach even in case of abnormal noise (Fig.7) i.e if the noise fluctuates violently in time, which would happen if the performance of the on-board sensors becomes bad (for exemple due to temperature effects). In all experiments, the bio-inspired neural model based EKF (B-EKF) was compared with the standard EKF algorithm (S-EKF). The

results showed that the localization error of the robot and the estimated error of the landmarks was less with B-EKF than with S-EKF in all experiments. Even in case of abnormal noise, the divergence problem of S-EKF was solved and its stability and accuracy were significantly improved.

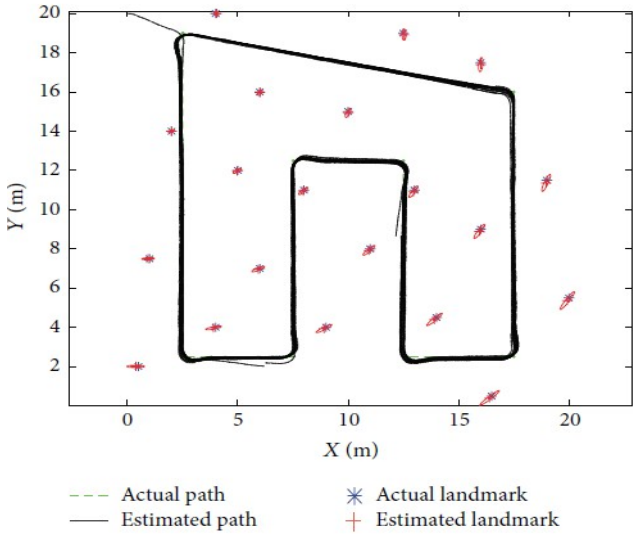


Fig.7. The results of the bio-inspired neurol model in a SLAM task in case of abnormal noise

Chapter 4

Comparison between standard and bio-inspired algorithms

In the last two chapters, we gave an overview of the most known techniques used in the two classes of probabilistic SLAM and bio-inspired SLAM. In this chapter, we make a comparison between these two classes and shed the light on the advantages and disadvantages of each one.
In order to achieve this, we start by presenting some of the validation methods used to compare different SLAM algorithms. Then we focus on the comparison in the second part of this chapter.

1. Validation methods used to compare SLAM algorithms

Validation methods are methods that aim at setting fix and precise criteria in order to evaluate the performance of a system. They enable us to realize an objective comparison based on these criteria.

The most used validation methods to compare SLAM algorithms are the following:

- **Localization error :** This error is usually evaluated as a standard deviation in function of the number of landmarks. It measures the accuracy of the algorithms and their liability in estimating the position of the robot.
- **Computation time :** It represents the time needed by the algorithm to achieve the goal of the SLAM problem. It can be measured for a particular step of the algorithm or for the whole mapping and localization purpose.
- **Computational cost :** It measures the complexity of the SLAM algorithm in terms of number of operations. This cost has a direct impact on the speed (computation time) of the SLAM algorithm. In case the cost is very high, it can exceed the power of the hardware used for the SLAM calculation.
- **Distance of loop-closing :** loop closing is the task of deciding whether or not the robot has, after an excursion a certain length, returned to a previously visited area. Reliable loop closing is both essential and hard in SLAM. Therefore the distance achieved before a correct loop-closing is an important index of the robustness and accuracy of the algorithm.

- **Scalability:** Localization and mapping in unknown environments becomes more difficult in large scale environments. In fact as the size of the environment gets bigger, the number of landmarks and therefore the computational cost of the SLAM algorithm usually increases. Hence the scalability or the ability of the robot to function well as the size of the mapped environment or the number of landmarks increases is an important criteria for estimating the performance of a SLAM algorithm.
- **Dynamism of the robot environment:** This criterion measures the ability of the robot to work in case of changes of location of other agents in the environment. This creates a big challenge for SLAM algorithms, because it introduces inconsistent sensor measurements.
- **Non-linear problem and non-gaussian noise:** Since most of the SLAM algorithm are base on the assumption that variables are linear and noise is gaussian, this criterion estimates the ability of the SLAM algorithm to work in a non-linear environment and with non-gaussian noise.
- **Robustness against noise and uncertainties:** A robust SLAM algorithm should be able perform well even in case of noisy data or ambiguous observations and high uncertainties. Robustness is an important criterion in SLAM because it is needed to obtain good results even with bad conditions.

2. Comparison of the two classes of SLAM algorithms

In this section, we establish a comparison between the two classes of probabilistic (or regular) and bio-inspired SLAM algorithms based on the validation methods presented in the previous section.

Firstly, concerning the dynamism of the robot environment, there are almost no regular SLAM algorithms that can deal with mapping in dynamic environments. Most of the probabilistic solutions to the SLAM problem aim at producing accurate maps of areas that are assumed to be static. There is a probabilistic SLAM algorithm called vSLAM which is able to work in a dynamic indoor environment. However, when tested in an outdoor environment, it could achieve only a distance of 200 m of mapping which is not comparable to the distances achieved by bio-inspired SLAM algorithms. The experiment of Milford and Wyeth in 2010 showed that RatSLAM system is capable of moving autonomously in a highly dynamic indoor environment. It was able to perform 1,143 delivery tasks in a working office environment and travel a total distance of more than 40 km over 37 hours with an accuracy over 98%.
Regular SLAM can not achieve this performance due to the important increase in complexity with the number of new landmarks that appear in a dynamic environment. This increasing complexity for probabilistic approaches with the number of landmarks is mainly the reason why bio-inspired SLAM algorithms have a better scalability and computation time than the regular ones. For example, the best mapping distance achieved by a regular SLAM algorithm so far was in Victoria Park with FastSLAM algorithm. FastSLAM was able to map an outdoor environment over a length of 3.5 km. On the other side, RatSLAM was successfully tested in an outdoor environment on a 66 km car journey through a complex suburban road network. Using only a web camera operating at 10 Hz, RatSLAM generated a coherent map of the entire environment at real-time speed. As far as we know, non of the leading state-of-the-art probabilistic

approaches was able to map the distance that RatSLAM achieved. It seems that almost all the regular SLAM algorithms can not perform consistent maps for very large areas , mainly due to the high increase of the computational cost and due to the uncertainties that become prohibitive when the scenario becomes larger. Some recent publications try to deal with this problem by using multiple maps, or sub-maps that are lately used to build a larger global map. However, this method called hierarchical SLAM, rely considerably on some data association assumptions and cannot be generalized for all scenarios.

Concerning the loop-closing problem, which is one of the biggest challenges for SLAM algorithms today, there are different approaches to deal with it. Some of the common probabilistic approaches use the estimates produced by the SLAM algorithm itself about the robot's position and landmarks locations for the detection of a potential loop closure. In order to achieve that, the algorithm simply performs a nearest neighbor statistical gate on the likelihood of the current visual measurements given map and pose estimates. This naive approach does not usually give good results in estimating the loop closure. This is mainly the case when there are significant errors in pose estimates, which is very likely to happen in case of a transit around a long loop. This is a very critical problem because if a loop is not detected, previously visited areas will be re-mapped by the robot but in the wrong global location, which results in a significant accumulation of errors.

Other techniques that can be integrated in a SLAM algorithm try to avoid the previous approach and detect a loop closure independently of the estimated map and vehicle position. This methods are therefore independent of the category of the SLAM algorithm (probabilistic or bio-inspired) in which they are integrated and allow a better robustness in detecting loop closures because they are independent of the internal SLAM state. One of these techniques is based on exploiting visual features with some reliable image-feature descriptors such as SIFT and on feature saliency using The Scale Saliency algorithm proposed by Kadir and Brady. A good matching between images in this case leads to a better loop-closing detection than the one obtained with the previous approach. This approach was successfully tested with laser-based regular SLAM to close loops of around 100 m in an indoor environment. However cameras can also be used instead of lasers to detect the loops. For example Davison et al. achieved a vision-based SLAM algorithm, called MonoSLAM, which used a single hand-held camera to achieve real-time reliable SLAM loop-closing in small indoor environments. This system, based on EKF, uses some assumptions about the motion dynamics of the camera and an active search for visual features to perform the loop detection. This work was later expanded by Carlos Estrada et al. to outdoor environments by using the hierarchical map approach in order to optimally close a loop of 250 m long. The hierarchical map method they used allowed them to maintain independence at the local level and consistency at the global level at a computational cost linear with the size of the loop. Another algorithm published by Cummins et al. in 2007, which was also independent of the pose-estimation and relied only on the captured images and on probabilistic methods, was able to achieve a loop closure over a distance of 1,6 km. This algorithm was linear in the number of landmarks and used the appearance of the captured image to learn and generate a probabilistic model that can decide whether two observations originate from the same location which the key to detecting loop closures. However there was also a successful algorithm called miniSLAM, which did not only rely on visual observations but also on odometry measurements to produce a metric map representation and achieve a loop closure of 1,4 km length. Similarly, vSLAM used odometry combined with the visual

scale-invariant feature transform (SIFT) and was successfully tested in a outdoor environment but only with a 200m long loop.

On the other hand, bio-inspired SLAM algorithms seem also to perform very well in detecting loop closure. For example, RatSLAM was able to map in real-time a loop of up to 5 km length using only visual sensory information. This system showed its high performance in closing large loops even in case of significant path integration errors and visually ambiguous environments, detecting even inner loops inside of an outer loop.

When it comes to computation time, both classes of SLAM were proven to be able of real-time SLAM. However, the problem with probabilistic approaches is that they can not scale to large environment even if they can perform a quick mapping and localization for small environments. As mentioned before, the reason for this bad scalability is the highly increasing complexity. For example the EKF-based SLAM maintains a full $N*N$ coavariance matrix for N landmarks. This covariance is updated with each measurement at $O(N^2)$ computation cost, which results in $O(N^2)$ complexity for the algorithm. Therefore, if real-time is desired for this EKF-based algorithm, the total number of landmarks should not exceed 100. Only a few improvements were achieved concerning the complexity of probabilistic. For example, an algorithm proposed by Ethan Eade and Tom Drummond, was able to reduce the complexity of the particle filter used by FastSLAM system from $O(N^2)$ to $O(N)$ However, this algorithm was based on the assumption that the pose of the robot is known, which results in the independence between the locations of the different landmarks. However in case of a large environment, the robustness of the algorithm depends not only on the number of landmarks, but also on the accumulation of error concerning the robot's pose. Therefore, this algorithm was successfully tested in a small indoor environment containing many landmarks but not in a large environment with a large loop closure as the pose error increases significantly in this case.

Concerning accuracy, probabilistic techniques are known to be able to produce high resolution, more accurate maps that represent faithfully the environment, such as the map represented in Fig.7. In this map achieved by Grisetti, Stachniss et al. in the Freiburg campus environment, a high resolution map was realized using the extended Rao-Blackwellized particle filter method.

Furthermore many probabilistic based mapping methods attempt to produce a Cartesian map with direct spatial correspondence to the physical environment. For example a 2.5 meter wide, 16.7 meter long corridor should appear as an identically sized region in the constructed map. This high degree of accuracy is however, most of the time, not very useful. In fact, for general navigation tasks like goal navigation, there is no need for a very detailed map as much as speed is needed. The problem with probabilistic methods is that for large environments, the amount of data stored in the map becomes quite large due to the high accuracy, and requires therefore more computation time. Place cells give on the contrary, a less precise map but that is sufficient for the navigation of the animal. They also don't provide a direct Cartesian correspondence between a map and the environment .

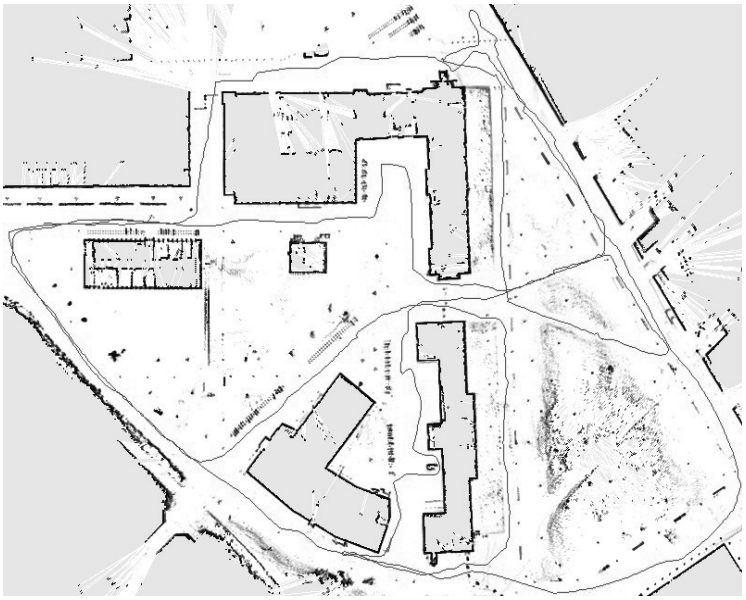


Fig.8. High resolution grid map of the University of Freiburg campus, measuring 250 × 250 m,with a grid resolution of 0.01 m. The map was generated using the extended Rao-Blackwellizedparticle filter method

The key strength of probabilistic techniques is perhaps their robustness and ability to deal with noise and ambiguity in robot sensors and environment. Therefore, in case of bad sensors, i.e with much noise and uncertainty, probabilistic algorithms usually give a more robust map than bio-inspired algorithms. Ambiguous features or landmarks in the environment can even become useful if they are well considered by the probabilistic algorithm rather than becoming failure points.

Finally, there is a problem which is intrinsically related to the probabilistic approaches. Many of therm are based on the assumption of linearity and Gaussian noise which is not always the case in the environment. This limits significantly the performance of these algorithms and requires some extensions and new solutions in order to be able to deal with all possible kinds of environment.

Chapter 5

Conclusion

We have presented in this report the two main classes of SLAM algorithms: probabilistic algorithms and bio-inspired algorithms. We tried to give for each class the main techniques and methods used and illustrate them with examples of real SLAM applications. We presented mainly FastSLAM, one of the most known probabilistic algorithms. We also presented RatSLAM and a neural bio-inspired model of extended Kalman filter as two successful examples bio-inspired algorithms. We also tried to establish a comparison between the two SLAM classes, based on some validation methods used to evaluate the performance of a SLAM algorithm. We have seen that probabilistic algorithms can provide us with very accurate and faithful algorithms to the real environment, but that this high accuracy has a high computational cost that makes them slower and more complex than bio-inspired algorithms especially for large-scale environments. We have also seen that the probabilistic approaches are more adapted to situations where there is an important amount of noise and uncertainty, but that the restrictions on the Gaussian model of noise and the linearity assumptions can significantly limit the use of probabilistic methods ans increase the localization and mapping errors. Finally, we have shown that for dynamic environments, bio-inspired are much more suited and can achieve very good results even in large areas.

The SLAM research community has certainly made good progress in the past years on the SLAM-estimation problem. However there are still many ways for possible improvements. For example, the SLAM scalability problem for arbitrarily large environments is still an open problem mainly due to the limitation in computer resources and to the increase in computational cost. Speeding up the SLAM algorithms in large-scale environments seems to be one of the biggest aims in the future. For probabilistic approaches mainly, there is still a long way to go in order to obtain more efficient and consistent SLAM in large scenarios and to solve the data association problem. For bio-inspired algorithms, despite the fact that they rely on bio-inspired models, many lessons are still to be learned from biology in order to make them more biologically realist and faithful to the real models.
Finally, most of the work done so far in SLAM literature focuses on ground and mainly indoor environments. Only few papers deal with SLAM in airborne applications and underwater conditions where they generally work with acoustic data not visual data. Therefore, this can also be a new research area for SLAM in the future.

List of Figures

Bibliography

Michael Montemerlo, Sebastian Thrun, Daphne Koller and Ben Wegbrei. FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem.

Josep AULINAS, Yvan PETILLOT, Joaquim SALVI and Xavier LLADÓ. The SLAM problem: a survey

Sebastian Thrun, Christian Martin, Yufeng Liu, Dirk Hahnel, Rosemary Emery-Montemerlo, Deepayan Chakrabarti and Wolfram Burgard. A Real-Time Expectation Maximization Algorithm for Acquiring Multi-Planar Maps of Indoor Environments with Mobile Robots.

Sebastian Thrun, Daphne Koller, Zoubin Ghahramani, Hugh Durrant-Whyte, and Andrew Y. Ng. Simultaneous Mapping and Localization With Sparse Extended Information Filters: Theory and Initial Results.

Matthew R. Walter, Ryan M. Eustice, and John J. Leonard. Exactly Sparse Extended Information Filters for Feature-Based SLAM.

Eric A. Wan and Rudolph van der Merwe. The Unscented Kalman Filter. Chapter 7

Zhenhe Chen, Jagath Samarabandu and Ranga Rodrigo. Recent advances in simultaneous localization and map-building using computer vision.

Michael J. Milford and Gordon F. Wyeth. Mapping a Suburb With a Single Camera Using a Biologically Inspired SLAM System.

Andrew J. Davison, Ian D. Reid, Nicholas D. Molton, and Olivier Stasse. MonoSLAM: real-time single camera SLAM.

John Kubie. Human Grid Cells, 2013. http://blog.brainfacts.org/2013/08/human-grid-cells/#.VZuSKPntmkp

Sebastian Thrun. Probabilistic Algorithms in Robotics , April 2000.

Sebastian Thrun. Robotic Mapping: A Survey, February 2002.

Dirk Hähnel, Wolfram Burgard, Dieter Fox and Sebastian Thrun. An Efficient FastSLAM Algorithm for Generating Maps of Large-Scale Cyclic Environments from Raw Laser Range Measurements.

Michael Montemerlo, Sebastian Thrun, Daphne Koller and Ben Wegbreit. FastSLAM 2.0: An Improved Particle Filtering Algorithm for Simultaneous Localization and Mapping that Provably Converges.

David Törnqvist, Thomas B. Schön, Rickard Karlsson and Fredrik Gustafsson. Particle Filter SLAM with High Dimensional Vehicle Model.

Giorgio Grisetti, Cyrill Stachniss and Wolfram Burgard. Improved Techniques for Grid Mapping with Rao-Blackwellized Particle Filters.

Thomas Schön and Fredrik Gustafsson. Marginalized Particle Filters for Mixed Linear/Nonlinear State-Space Models.

Guivant Jose. Efficient simultaneous localization and mapping in large environments. Chapter 1.

Hugh Durrant-Whyte and Tim Bailey. Simultaneous Localisation and Mapping (SLAM): Part I The Essential Algorithms and Part II.

Gerasimos G. Rigatos. Extended Information Filtering and nonlinear control for cooperating robot harvesters.

L.M. Paz, P. Jensfelt, J.D. Tard´os and J. Neira. EKF SLAM updates in $O(n)$ with *Divide and Conquer* SLAM.

Sebastian Thrun. A Probabilistic Online Mapping Algorithm for Teams of Mobile Robots.

Wolfram Burgard, Dieter Fox, Hauke Jans, Christian Matenar and Sebastian Thrun. Sonar-BasedMapping WithMobile Robots Using EM.

Matthew R. Walter, Ryan M. Eustice, and John J. Leonard. Exactly Sparse Extended Information Filters for Feature-Based SLAM.

Niko Sünderhauf, Sven Lange and Peter Protzel. Using the Unscented Kalman Filter in Mono-SLAM with Inverse Depth Parametrization for Autonomous Airship Control.

Laura A. Clemente, Andrew J. Davison, Ian D. Reid, José Neira and Juan D. Tard´os. Mapping Large Loops with a Single Hand-Held Camera.

Jose Guivant and Eduardo Nebot. Improving Computational and Memory Requirements of Simultaneous Localization and Map Building Algorithms.

Kai Xiong, Hongyue Zhang and Liangdong Liu. Adaptive Robust Extended Kalman Filter.

Simon Haykin. Kalman Filtering and Neural Networks. ISBNs: 0-471-36998-5 (Hardback); 0-471-22154-6 (Electronic).

Theodore S. Lindsey. On the Kalman Filter and Its Variations. April 23, 2014.

Josep Aulinas, Xavier Llad´o, Joaquim Salvi and Yvan R. Petillot. SLAM based Selective Submap Joining for the Victoria Park Dataset.

Henrik Andreasson, Tom Duckett and Achim Lilienthal. Mini-SLAM: Minimalistic Visual SLAM in Large-Scale Environments Based on a New Interpretation of Image Similarity.

Denk C., Llobet-Blandino F., Galluppi F., Plana LA., Furber S., and Conradt, J. (2013) Real-Time Interface Board for Closed-Loop Robotic Tasks on the SpiNNaker Neural Computing System, International Conf. on Artificial Neural Networks (ICANN), p. 467-74, Sofia, Bulgaria.

Stephen Se*, Member*, David G. Lowe and James J. Little. Vision-Based Global Localization and Mapping for Mobile Robots.

Hoffmann R., Weikersdorfer D., and Conradt J. (2013) Autonomous Indoor Exploration with an Event-Based Visual SLAM System, European Conference on Mobile Robots, p. 38-43, Barcelona, Spain.

Matthew Walter*,* Ryan Eustice and John Leonard. A Provably Consistent Method for Imposing Sparsity in Feature-based SLAM Information Filters.

Stephen Se, David Lowe and Jim Little. Mobile Robot Localization and Mapping with Uncertainty using Scale-Invariant Visual Landmarks.

M.W.M.G. Dissanayake, P.Newman, S. Clark, H.F. Durrant-Whyte and M. Csorba. A Solution to the Simultaneous Localisation and Map Building (SLAM) Problem.

A. C. Murillo, J. J. Guerrero and C. Sagüés. SURF features for efficient robot localization with omnidirectional images.

Sebastian Thrun and Yufeng Liu. Multi-Robot SLAM With Sparse Extended Information Filers.

Eric A. Wan and Rudolph van der Merwe. The Unscented Kalman Filter for Nonlinear Estimation.

Niklas Karlsson, Enrico Di Bernardo, Jim Ostrowski, Luis Goncalves, Paolo Pirjanian, and Mario E. Munich. The vSLAM Algorithm for Robust Localization and Mapping.

Zsolt Kira. Evaluation of VSLAM in Outdoor environments. November 2014.

Jeffrey S. Taube, Robert U. Muller, and James B. Ranck, Jr. Head-Direction Cells Recorded from the Postsubiculum in Freely Moving Rats. I. Description and Quantitative Analysis.

Stefan Leutgeb, Jill K. Leutgeb, Alessandro Treves, May-Britt Moser and Edvard I.Moser. Distinct Ensemble Codes in Hippocampal Areas CA3 and CA1.

Alejandra Barrera and Alfredo Weitzenfeld. Biologically-inspired Robot Spatial Cognition based on Rat Neurophysiological Studies.

Jianjun Ni, ChuWang, Xinnan Fan, and Simon X. Yang. A Bioinspired Neural Model Based Extended Kalman Filter for Robot SLAM.

Brett Browning. Biologically Plausible Spatial Navigation for a Mobile Robot. Department of Computer Science and Electrical Engineering.The University of Queensland.

Niko Sünderhauf and Peter Protzel. Learning from Nature: Biologically Inspired Robot Navigation and SLAM-A Review.

P.Gaussier, A.Revel, J.P.Banquet and V.Babeau. From view cells and place cells to cognitive map learning : processing stages of the hippocampal system.

Niko Sünderhauf, Peer Neubert and Peter Protzel. The Causal Update Filter – A Novel Biologically Inspired Filter Paradigm for Appearance-Based SLAM.

Torkel Hafting, Marianne Fyhn, Sturla Molden, May-Britt Moser and Edvard I. Moser. Microstructure of a spatial map in the entorhinal cortex.

Robert C. Wilson and Leif H. Finkel. A Neural Implementation of the Kalman Filter.

Vegard H. Brun, Mona K. Otnaess, Sturla Molden, Hill-Aina Steffenach, Menno P. Witter, May-Britt Moser and Edvard I. Moser. Place Cells and Place Recognition Maintained by Direct Entorhinal-Hippocampal Circuitry.

David Prasser, GordonWyeth and Michael Milford. Experiments in Outdoor Operation of RatSLAM.

Michael Milford and Gordon Wye. Spatial Mapping and Map Exploitation: A Bio-inspired Engineering Perspective.

Marianne Fyhn, Sturla Molden, Menno P. Witter, Edvard I. Moser and May-Britt Moser. Spatial Representation in the Entorhinal Cortex.

Paul Newman and Kin Ho. SLAM- Loop Closing with Visually Salient Features.

Conradt, J., Tevatia, G., Vijayakumar, S., & Schaal, S. (2000). On-line Learning for Humanoid Robot Systems, International Conference on Machine Learning (ICML2000), Stanford, USA.

Weikersdorfer D., Adrian DB., Cremers D., and Conradt J. (2014) Event-based 3D SLAM with a depth-augmented dynamic vision sensor, IEEE ICRA 2014, Int. Conf. on Robotics and Automation, Hong-Kong, China, June 2014.

Weikersdorfer D., Hoffmann R., and Conradt J. (2013) Simultaneous Localization and Mapping for event-based Vision Systems, International Conference on Computer Vision Systems (ICVS), p. 133-142, St. Petersburg, Russia.

Weikersdorfer D., Conradt J. (2012), Event-based Particle Filtering for Robot Self-Localization, Proceedings of the IEEE International Conference on Robotics and Biomimetics (IEEE-ROBIO), pages: 866-870, Guangzhou, China.

License