AN INTRODUCTORY OVERVIEW TO SLAM ALGORITHMS

Rishab Shah

4/5/2017

Boston University

Department of Electrical and Computer Engineering

Technical Report No. ECE-

BOSTON UNIVERSITY

AN INTRODUCTORY OVERVIEW TO SLAM ALGORITHMS

Rishab Shah

Boston University
Department of Electrical and Computer Engineering
8 Saint Mary's Street
Boston, MA 02215
www.bu.edu/ece

4/5/2017

Technical Report No. ECE-

Contents

1 Introduction **1**
1.1 Loop closures 1

2 The SLAM process **3**

3 State Estimation and State Update **3**
3.1 Extended Kalman Filter (EKF) SLAM 4
3.2 Rao-Blackwellized Particle Filter 6

4 Feature Extraction **7**

5 Data Association **8**
5.1 SIFT 8

6 History and the state-of-the-art in SLAM **9**
6.1 Monocular vs. Stereo based SLAM 10
6.2 Sparse feature based vs. Dense feature based SLAM 10
6.3 Continous-time SLAM 10

7 Conclusion **11**

List of Figures

1 Example of loop closure [4] 2
2 Spring network analogy. The landmarks are connected by springs describing correlations between landmarks. As the vehicle moves back and forth through the environment, spring stiffness or correlations increase (red links become thicker). As landmarks are observed and estimated locations are corrected,these changes are propagated through the spring network. Note, the robot itself is correlated to the map. [1] 2
3 EKF SLAM simulation 5
4 Plot of position error over time and position uncertainty over time. . 5
5 A single realization of robot trajectory in the FastSLAM algorithm. The ellipsoids show the proposal distribution for each update stage, from which a robot pose is sampled, and, assuming this pose is perfect, the observed landmarks are updated. Thus, the map for a single particle is governed by the accuracy of the trajectory. Many such trajectories provide a probabilistic model of robot location. [2] 7
6 After scale space extrema are detected (their location being shown in the uppermost image) the SIFT algorithm discards low contrast keypoints (remaining points are shown in the middle image) and then filters out those located on edges. Resulting set of keypoints is shown on last image. 9

1 Introduction

The simultaneous localization and mapping (SLAM) problem asks the question "Is it possible for an autonomous vehicle to start in an unknown environment and then to incrementally build a map of this environment while simultaneously using this map to compute the vehicle pose?". This ability to place an autonomous vehicle at an unknown location in an unknown environment and then have it build a map, using only relative observations of the environment, and then to use this map simultaneously to navigate would indeed make such a robot "autonomous". SLAM eliminates the need for *a priori* knowledge of the topology of the surroundings. SLAM is today routinely achieved in robot systems using advanced methods of sequential Bayesian inference and SLAM algorithms are starting to cross over into practical systems.
SLAM has been formulated and solved as a theoretical problem in a number of different forms. SLAM has also been implemented in a myriad of fields from indoor robots to outdoor, underwater, and airborne systems. However, many significant issues remain in practically realizing more general SLAM solutions and more importantly in building and perceptually rich maps as a part of a SLAM algorithm.

1.1 Loop closures

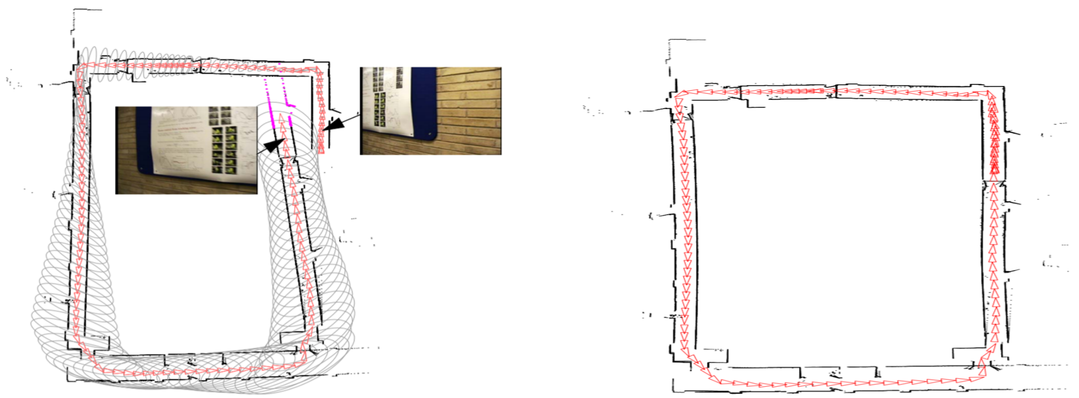
Loop closures consist of recognizing a place that has already been visited in an excursion of arbitrary length. Problems arise when two different place from the environment are recongnized as the same, this is known as *perceptual alisaing*. Detection methods for loop closure detection can be divided into three categories: map-to-map, image-to-image and image-to-map. Categories differ mainly from where the association data are taken from. However, the ideal would be to build a system that combines the advantages of all three categories. Newman et. al [3] propose a system where they use the notion of visual saliency to focus the selection of sutiable image-feature descriptors for storage in a database.This time information is used to discover loop closing events, and this is achieved independently of estimated pose and the map. All uncertainties collapse after a loop closure whether or not the loop closure performed was correct or not.

In conclusion, a good real-time SLAM algorithm has to provide:

1. Corresponding observations of scene features among a subset of keyframes, where keyframes are single frames in an animated sequence that occur at an important point in that sequence.

2. As complexity grows with the number of keyframes, their selection should avoid redundancy.

3. A strong network of keyframes and points to provide accurate results, i.e., well-spread and good loop closure matches.



(a) Robot pose uncertainty before loop closure.

(b) Robot pose uncertainty after loop closure.

Figure 1: Example of loop closure [4]

4. An initial esitmation of keyframe poses and points for non-linear optimization.

5. A local map in exploration where optimization is focused to achieve scalability.

6. The ability to perform fast global optimizations to close loops in real time.

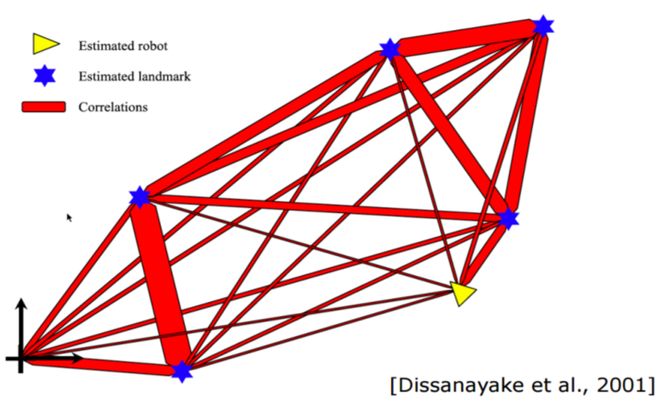


Figure 2: Spring network analogy. The landmarks are connected by springs describing correlations between landmarks. As the vehicle moves back and forth through the environment, spring stiffness or correlations increase (red links become thicker). As landmarks are observed and estimated locations are corrected,these changes are propagated through the spring network. Note, the robot itself is correlated to the map. [1]

2 The SLAM process

The important aspects of a SLAM algorithms consist of five steps mainly:

1. Landmark Extraction
2. Data association
3. State estimation
4. State update
5. Landmark Update

There is, in most algorithms, no clear distinction between these steps. Landmarks are features which can be easily reobserved and distinguished from the environment. It is helpful for the computer or the SLAM process if these *landmarks* are plentiful and stationary. Landmark extraction can be performed by two general metholds: Spike landmarks: where these are threshold values that exceed certain values between two measurements (fails in smooth enviroments), or RANSAC (Random Sampling and Consensus): where we extract lines using least squares approximation. This method (RANSAC) does not pick people as landmarks because they do not have the characterstic of the shape of a line. Range measurement is usually performed by laser scanners, vision (cameras, infrared cameras, etc.) or SONAR. The problem is that in most cases, laser odometry data matching is a complicated process and this takes up a lot of the computation of the processor which in turn increases the computation complexity of the SLAM process which was already a computation heavy procedure.

3 State Estimation and State Update

In the past, some of the methods used to perform state estimation and update are:

- Extended Kalman Filter
- Bayes filter
- Particle filters
- Rao-Blackwellized Particle Filter

The most common representation is in the form of a state-space model with additive Gaussian noise, leading to the use of the extended Kalman filter (EKF) to solve the SLAM problem. One important alternative representation is to describe the vehicle motion model as a set of samples of a more general nonGaussian probability distribution. This leads to the use of the Rao-Blackwellized particle filter, or FastSLAM algorithm, to solve the SLAM problem. While EKF-SLAM and FastSLAM are the two most important solution methods, newer alternatives, which offer much potential, have been proposed, including the use of the information-state form.

3.1 Extended Kalman Filter (EKF) SLAM

The basis for EKF SLAM method is to describe the vehicle motion in the form where the Robot controls are:

$$u_{1:k} = \{u_1, u_2, u_3.., u_k\} \tag{1}$$

and the observations are :

$$z_{1:k} = \{z_1, z_2, z_3..., z_k\} \tag{2}$$

The basic algorithm is given by two steps, i.e, the predict step and the update step:

$$\hat{\mu} = g(u_k, \mu_{k-1}) \tag{3}$$

where $\hat{\mu}$ is the forecasted model's state and $g(.)$ is the state equation,

$$\hat{\Sigma}_k = G_k \Sigma_{k-1} G_k^T + R_k \tag{4}$$

and this is the covariance of the state estimate. Both the state and the covariance are assumed to have a additive, Gaussian distributed noise with covariance Q and R respectively. G is the observation matrix and relates the output of the sensor to the state vector when observing the k-th landmark. The update step is given by

$$K_k = \hat{\Sigma}_k H_k^T (H_k \hat{\Sigma}_k H_k^T + Q_k)^{-1} \tag{5}$$

where K_k is the Kalman gain that is responsible for estimting the correct state of the dynamic system.

$$\mu_k = \hat{\mu}_k + K_k(z_k - h(\hat{\mu}_k)) \tag{6}$$

$$\Sigma_k = (I - K_k H_k)\hat{\Sigma}_k \tag{7}$$

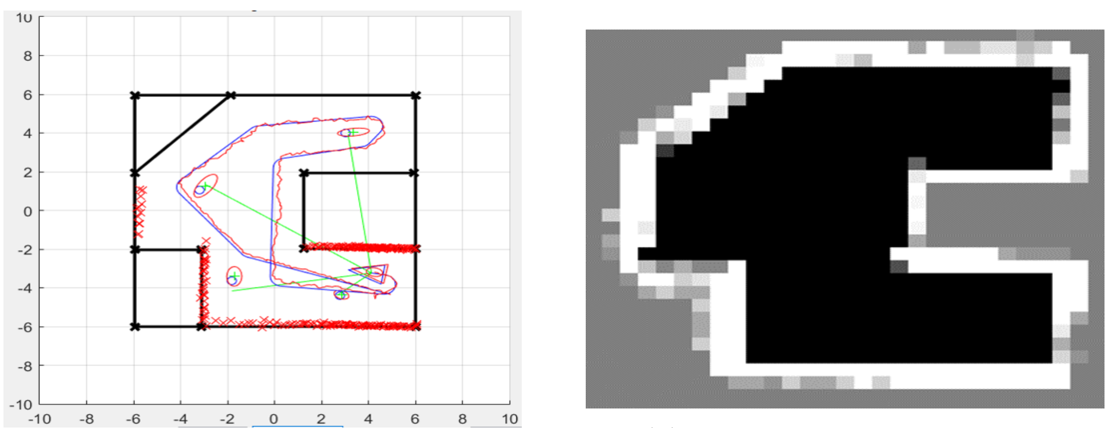
So what happens here basically in the correction step is:

- The data association takes place using advanced image processing techniques that make use of feature descriptors and the measurement observes the landmark with index k.
- The landmark is initialized if it was previously unobserved.
- The expected observation is computed.
- The Jacobian of $h(.)$ is computed.
- The Kalman gain is computed.

It is important to note that the measurement step is completed in a single step and requies only one full belief update, all the angular components need to normalized. The determinant of any sub-matrix of the map covariance matrix decreases monotonically. and new landmarks are initialized with maximum uncertainty. The EKF SLAM becomes computationally intractable for large maps. In the EKF-SLAM problem, convergence of the map is manifest in the monotonic convergence of the determinant of the map covariance matrix and all landmark pair submatrices, toward zero. The individual landmark variances converge toward a lower bound determined by initial uncertainties in robot position and observations. Convergence and consistency can only be guaranteed in the linear case.

3.1.1 Simulation

A simple implementation of the EKF SLAM is performed by defining the map in the simulation environment, providing artificial landmarks and waypoints for the "robot" to traverse through and the method is implemented to check different graphs such as the uncertainity over time and position uncertainty over time in the x and the y-direction where we can see large spikes in the data due to lack of landmarks near the end of the run and uncertainties collapsing due to the loop closing event.



(a) Simulation of EKF SLAM in a room

(b) Generated map after simulation.

Figure 3: EKF SLAM simulation

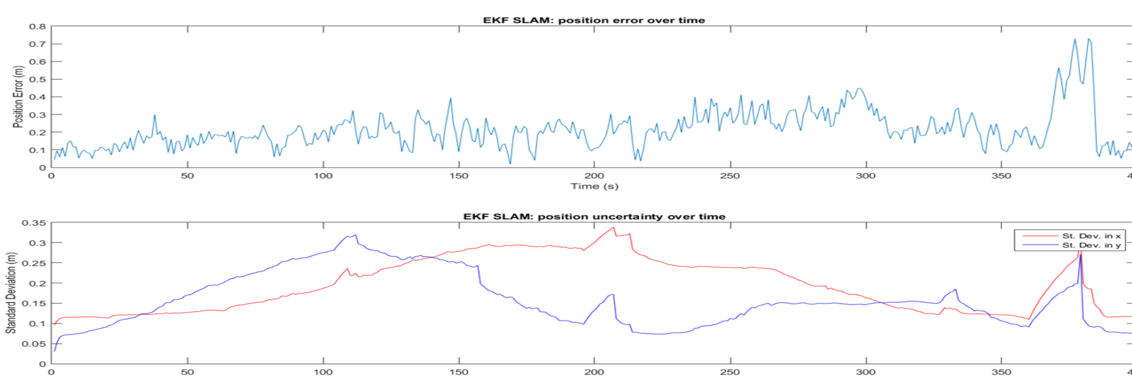


Figure 4: Plot of position error over time and position uncertainty over time.

3.2 Rao-Blackwellized Particle Filter

The FastSLAM algorithm, marked a fundamental conceptual shift in the design of recursive probabilistic SLAM. Previous efforts focused on improving the performance of EKF-SLAM, while retaining its essential linear Gaussian assumptions. FastSLAM, with its basis in recursive Monte Carlo sampling, or particle filtering, was the first to directly represent the nonlinear process model and non-Gaussian pose distribution. (FastSLAM still linearizes the observation model, but this is typically a reasonable approximation for range-bearing measurements when the vehicle pose is known.) The Rao-Blackwellized particle filter (RBPF) is a non-parametric, recursive type of Bayes filter. The noise models and the other modelling is not limited to Gaussian distributions. This filter works very well in low dimensional spaces. Also, this method uses Monte-Carlo localization, which is the standard for mobile robot localization these days.
The joint SLAM state may be factored into a vehicle component and a conditional map component:

$$P(\mathbf{X}_{0:k}, \mathbf{m}|\mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_0) = P(\mathbf{m}|\mathbf{X}_{0:k}, \mathbf{Z}_{0:k})P(\mathbf{X}_{0:k}|\mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_0) \tag{8}$$

This describes the joint posterior density of landmark locations and vehicle state (at time k), given recorded observations and control inputs together with the initial state. Similar to EKF SLAM, $\mathbf{X}_{0:k}$ is the state, and $\mathbf{Z}_{0:k}$ is the observation of the environment and $\mathbf{x}_0$ is the initial condition. These Monte-Carlo methods , especially particle filters, predate the existence of the Extended Kalman filter bur are more computationally expensive for any moderately dimensioned state space. The general form of RBPF is as follows.
We assume that at time $k-1$, the joint state is represented by
$\{\mathbf{w}^{(i)}_{k-1}, \mathbf{X}^{(i)}_{k-1}, P(\mathbf{m}|\mathbf{X}^{(i)}_{k-1}, \mathbf{Z}^{(i)}_{k-1})\}^N_i$.

1. For each particle, compute a proposal distribution and draw a sample from it. This new sample is joined to the particle history.

2. Weight samples according to a importance function that uses the observation model and the motion model.

3. Perform resampling, either after every time-step or once the weight varies a threshold. Resampling is achieved using select particles, with replacement, including their associated maps with probability of selection proportional to sampled weight.

4. These selected particles are given a uniform weight.

5. For each particle, perform an EKF update on the observed landmarks as a mapping operation with the known vehicle pose.

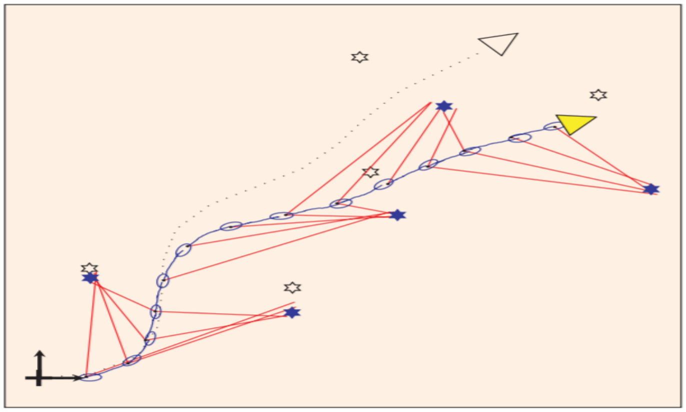


Figure 5: A single realization of robot trajectory in the FastSLAM algorithm. The ellipsoids show the proposal distribution for each update stage, from which a robot pose is sampled, and, assuming this pose is perfect, the observed landmarks are updated. Thus, the map for a single particle is governed by the accuracy of the trajectory. Many such trajectories provide a probabilistic model of robot location. [2]

4 Feature Extraction

One of the classic methods for feature extraction is the Random Sampling and Consensus (RANSAC), since it becomes a model fitting problem now, and RANSAC is a very good and robust model fitting procedure. Before delving into the algorithm for RANSAC, let's tkae a look a term will be used often in the algorithm, i.e., an *outlier*. A variation of the definition for an outlier is that "A data point is considered to be an outlier if it will not fi the "true" model instantiated by the "true" set parameters within some error threshold that derives maximum deviation attributable to the effects of noise." A *breakdown point* represents minimum fraction of outliers that are suuficient to produce an arbitrarily large bias. The assumptions are RANSAC are as follows.

- Parameters can be estimated from N data items.
- There are M data items in total.
- The probability of a randomly selected data item being a part of the good model is P_g.
- The probability that the algorithm will exit without finding a good fit if one exists is P_{fail}.

The algorithm itself is given by:

1. Selects N data points at random.
2. Estimates parameter $\vec{x}$.

3. Finds how many data items (of M) fit the model with the parameter vector $\vec{x}$ within a user given tolerance K.

4. If K is big enough, accept fit and exit.

5. Repeat steps 1-4 L times.

6. Fail if you get here.

K depends on percentage of data we think belongs to the structure and how many structures are in the image. For multiple structures, after success, remove the fit and redo RANSAC. An advantage of RANSAC is that it gives a high degree of accuracy even with a lot of outliers in the data set. A disadvantage of this method is that there is no upper bound on the time to compute these parameters, and its performance is bad when no inliers are less than 50%.

5 Data Association

Data association is performed by using advanced image processing techniques to "read" landmarks from the video each frame and give them weights based on different methods. Some of the popular data association mehtods in use today in the SLAM field are:

- Histogram of Oriented Gradients (HOG)
- Laplacian of Gaussian (LOG)
- Scale Invariant Feature Transform (SIFT) [5] [6]
- Speeded Up Robust Features (SURF)
- Binary Robust Independent Element Features (BRIEF)
- Oriented Fast and Rotated BRIEF (ORB)
- Binary Robust Invariant Scalable Keypoints (BRISK)
- Fast Retina Keypoint (FREAK)

5.1 SIFT

For any object in an image, interesting points on the object can be extracted to provide a "feature description" of the object. This description, extracted from a training image, can then be used to identify the object when attempting to locate the object in a test image containing many other objects. To perform reliable recognition, it is important that the features extracted from the training image be detectable even

under changes in image scale, noise and illumination. Such points usually lie on high-contrast regions of the image, such as object edges.
Another important characteristic of these features is that the relative positions between them in the original scene shouldn't change from one image to another. For example, if only the four corners of a door were used as features, they would work regardless of the door's position; but if points in the frame were also used, the recognition would fail if the door is opened or closed. Similarly, features located in articulated or flexible objects would typically not work if any change in their internal geometry happens between two images in the set being processed. However, in practice SIFT detects and uses a much larger number of features from the images, which reduces the contribution of the errors caused by these local variations in the average error of all feature matching errors. SIFT can robustly identify objects even among clutter and under partial occlusion, because the SIFT feature descriptor is invariant to uniform scaling, orientation, illumination changes, and partially invariant to affine distortion.

Figure 6: After scale space extrema are detected (their location being shown in the uppermost image) the SIFT algorithm discards low contrast keypoints (remaining points are shown in the middle image) and then filters out those located on edges. Resulting set of keypoints is shown on last image.

6 History and the state-of-the-art in SLAM

SLAM algorithms have been continously evolving since its inception and has gone through many types of methodologies that can be used to solve the seemingly complex problem of simultaneously tracking and mapping in an unknown environment for robotics. The early stages started with deterministic SLAM methods that used the Kalman filter for the state estimation, which proved to be very inaccurate dure its application of a linear model on a non-linear system. Then models started to improve with the EKF SLAM being used which was still a linearization of the system but showed better results. Then, researchers moved to probabilistic methods to perform state estimation, and as we saw previously, it moved to particle filters which were very successful and newer variants of these filters are currently in use. In the past, the computation power that was available on-board the processor of the robot was not good enough in addition to the absence of complex methods to produce and analyze 3D maps, and hence 2D SLAM was used and computed. The state-of-the-art algorithms almost all them use better processors that can perform the SLAM computation on-board or online and compute 3D maps of the environment.

6.1 Monocular vs. Stereo based SLAM

There are still debates about the methodologies in the new "era" of SLAM algorithms of using monocular systems instead stereo sesning systems. Each of them have their perks, some of them for monocular being that cost of the system is reduced while at the same time, increasing computation complexity due to more work that needs to be done to get the same amount performance out the SLAM algorithms.

6.2 Sparse feature based vs. Dense feature based SLAM

Some other types of systems that need some serious consideration are the use of sparse feature based methods against dense feature/ direct methods. The advatnages for dense feature based SLAM algorithms are that they do not need feature extraction and avoid corresponding artifacts. Also,they are more robust to blur, low texture and high frequency environments (e.g., Asphalt). Some disadvantages are theat they assume surface reflectance model that in real scenes produces its own artifacts. These algorithms are also affected by rolling shutter, auto-gain and auto-exposure. Also, they are computationally demanding.
In contrast, sparse feature based methods match features with a wide baseline because of good invariance to viewpoint and illumination changes. "Bundle adjustment" jointly optimizes camera poses over sensor adjustments. Bundle adjustments provide accurate estimates of camera localizations as well as a sparse geometrical representation, given that a strong network of matches amd good inital guesses are provided. Finally, sparse feature based SLAM methods are more accurate in real time.

6.3 Continous-time SLAM

All methods described up until now have been using discrete measurements. One state-of-the-art SLAM approach is to investigate the problem of planning under uncertainty for robots.The planning approach produces smooth and natural trajectories and is able to impose soft upper bounds on the uncertainty. They expoit the results of this analysis to identify current limitations and show that the framework can accomodate several extensions [8].

7 Conclusion

Revisiting the question “Is SLAM necessary?”, it has to be said that the answer depends on the application, but quite often the answer is affirmative. SLAM techniques will be increasingly relied upon to provide reliable metric in situations where solutions such as Global Positioning System (GPS) are unavailable or do not provide adequate accuracy. Online SLAM can also be used in self-driving cars where precision accuracy in localization is a requirement and is performed by matching current sensor data to a high definition map of the environment in advance. Another question that is asked more often thatn not is “Is SLAM solved?”, we can argue that this question cannot be answered without specifying a robot/environment/perfoormance combination.To achieve truly robust tracking and navigation for autonomous robots, more research is clearly needed.
Another fundamental question regards the design of metric and semantic representations for the environment. Despite the fact that the interaction with the environment is important for most applications of robotics, modern SLAM systems are not able to provide a tightly-coupled high-level understanding of the geometry and the semantic of the environment; the design of such representations must be task-driven.

References

[1] M. W. M. G. Dissanayake and P. Newman and S. Clark and H. F. Durrant-Whyte and M. Csorba, “A solution to the simultaneous localization and map building (SLAM) problem,” *IEEE Transactions on Robotics and Automation*, vol.17, pp.229-241, 2001.

[2] H. Durrant-Whyte and T. Bailey, “Simultaneous localization and mapping: part I”, *IEEE Robotics Automation Magazine*, vol. 13, pp 99-110, 2006.

[3] P. M. Newman, “On the solution to the simultaneous localization and map building problem”, *Ph.D. dissertation* , Dept. Mechan. Eng., Australian Centre for Field Robotics, Univ. Sydney, Sydney, Australia, 1999.

[4] P. Newman and Kin Ho, “SLAM-Loop Closing with Visually Salient Features,” *IEEE International Conference on Robotics and Automation*, pp. 635-642., 2005.

[5] D. Lowe, “Object recognition from local scale-invariant features.”, *Proc. 7th International Conference on Computer Vision*, pp. 1150-1157, 1999.

[6] D. Lowe, "Distinctive image features from scale-invariant key points.", *International Journal of Computer Vision*, vol. 60, pp. 91-110, 2004.

[7] C. Cadena and L. Carlone and H. Carrillo and Y. Latif and D. Scaramuzza and J. Neira and I. Reid and J.J. Leonard, "Past, Present, and Future of Simultaneous Localization And Mapping: Towards the Robust-Perception Age", *IEEE Transactions on Robotics*, vol. 32, pp. 1309-1332, 2016.

[8] V. Indelman, L. Carlone, F. Dellaert, "Planning in the continuous domain: A generalized belief space approach for autonomous navigation in unknown environments", *The International Journal of Robotics Research*, vol. 34, pp. 849-882, 2015.