# Enhancing Robustness and Accuracy of 3D SLAM Algorithm Using Dempster-Shafer Theory

Tatiana Berlenko
JETBRAINS LIMITED
Paphos, Cyprus
tatiana.berlenko@gmail.com

Anton Filatov
Saint-Petersburg Electrotechnical University "LETI"
St. Petersburg, Russia
aifilatov@etu.ru

***Abstract*—Simultaneous Localization and Mapping (SLAM) is a fundamental challenge in robotics, enabling autonomous navigation for robots in unknown environments. This article explores various approaches and methods for solving SLAM in both 2D and 3D cases. This article aims to show the importance of uncertainty measurement and control and suggest possible solutions and discuss future research directions in extending algorithms like VinySLAM [1] to the 3D domain.**

## I. Introduction

Localization is one part of the robot's navigation challenge. Localization is the determination of a robot's position in its environment. Autonomous navigation can allow robots to navigate in an unknown environment and perform tasks without human intervention. The solution of the SLAM (Simultaneous Localization and Mapping) problem will allow the robot to accurately determine its position on the map, and as a consequence, it would provide the means to make a robot truly autonomous [2].

There are different approaches to solving this problem, none of which is optimal. The choice of a particular SLAM algorithm, the choice of map representation, and the robot's representation of its belief about its position on the map affect the accuracy and speed of the final solution, as well as the amount of resources required.

The purpose of this paper is to show the importance of uncertainty measurement and control in the SLAM problem and its crucial influence in the 3D case. This article describes the beginning of the research of creating a 3D SLAM algorithm using implementation of Dempster-Shafer Theory.

## II. Problem Statement

At the moment there are many different approaches for SLAM in the 2D case.

The 3D case differs from the 2D case by adding one more dimension and, as a consequence, by increasing the amount of memory for storage and processing time of the map. This creates additional technical difficulties, which require the development of new algorithms and methods to effectively solve the problem of 3D SLAM. Consider some existing approaches.

### *A. Graph-based SLAM*

Graph SLAM represents the map and robot locations as a sparse graph. A graph is constructed with nodes representing robot poses or landmarks, and edges between nodes encoding sensor measurements that constrain the connected poses. However, due to noise in observations, these constraints may be contradictory. The main challenge in graph-based SLAM is to find a configuration of the nodes that is maximally consistent with the measurements, which involves solving a large error minimization problem. [3]

In Graph SLAM, loops often occur when the robot passes through the same area several times. These loops can lead to an accumulation of errors in the evaluation of the robot's trajectory and the environment map.

The method called Loop Closure Detection is used to solve this problem. This method detects when a robot passes through the same area multiple times and includes this information in the graph.

This algorithm requires comparing two nodes in the graph, but since it is unknown which nodes need to be compared, the worst case SLAM without optimizations requires to compare all nodes with all nodes. This operation is already expensive in the two-dimensional case, so is not done at every iteration of the algorithm. In the three-dimensional case, this operation becomes even more expensive. The 2D case uses three degrees of freedom: x, y, and rotation angle. In the 3D case, the number of degrees of freedom increases to six, including x, y, z coordinates and three rotation angles in each axis. This complicates the graph matching and optimization process, and may require additional computational resources and time to efficiently solve the problem.

### *B. Particle filter*

The particle filter is another way to solve the SLAM problem. In general, in the particle filtering algorithm, the posterior density of the distribution function is represented by a set of random particles with weights, each of which specifies a hypothetical position of the robot in space. Each such particle is evaluated for correspondence between the measurements from the sensors and the expected values that the robot should have received while in the given position and orientation. Based on the weight of the particles, the particle filter determines the most likely location of the robot. Particles that turn out to be less likely are discarded, and the most likely particles are multiplied.

There are several varieties of the particle filtering algorithm.

One of these families of algorithms, Rao–Blackwellized particle filters has a problem that lies in algorithmic complexity, measured in terms of the number of particles required to learn

an accurate map. Either reducing this quantity or improving the algorithm so that it is able to handle larger sample sets [4].

### C. Map Representation

One of the ways to represent the environment is using grid-based maps. The Probability Grid Map for the 2D case is a two-dimensional array of cells, each of which represents the probability that a given area of space is occupied by an obstacle. One of the variations of the grid-maps, Occupancy Grid Map, stores in every cell 0 or 1 in case this cell has an obstacle or not.

FastSLAM, one of the implementations of Particle filter, uses Occupancy Grid Map. One of graph-based SLAM, Cartographer [5], uses Probability Grid when building submaps.

Correspondingly, when using an Occupancy Grid Map or Probability Map for a 3D map, such a map is already a three-dimensional array. In [6] it is pointed out that at each step of prediction and correction of the robot's position all cells of grid-map are updated, and if the number of cells in the map is too large, the calculations may become too heavy to work in real time. As an example, the authors consider a 30x30 m environment with a cell size of 0.1mx 0.1mx1°. In this case, the number of cells that need to be updated at each step would be 30x30x100x360=32.4 million cells.

One possible solution in this case is to sacrifice accuracy and choose a larger mesh size. Another proposed solution is to use adapted cell decomposition, that is, cells of different sizes depending on the robot's confidence in its position. This implementation is more complex than cell size is fixed and, depending on the environment, can also be computationally demanding.

When creating grid-maps, there can be uncertainty about the state of the cell. For example, if a lidar beam pierced a cell the same number of times and encountered an obstacle, it is impossible to say unambiguously whether the cell in question is free or not. Another example, when a lidar beam cannot reach some space in the environment and cannot provide information about it.

### D. VINYSLAM

VinySLAM is a modification of the one-hypothesis tinySLAM [7] algorithm that uses the Transferable Belief Model (TBM) [8] which is the interpretation of the Dempster-Shafer Theory [9] to represent cell states in the Occupancy Grid Map. A cell can be in one of four states: occupied, empty, unknown, and conflict.

The algorithm is less robust than Grid-based FastSLAM (GMapping) and Graph SLAM (Cartographer) implementations. At the same time, based on measurement results, vinySLAM has accuracy close to Cartographer, as well as a much simpler implementation. VinySLAM can be found here [1].

Future research intends to extend VinySLAM to the three-dimensional case.

## III. Applying the TBM and the Dempster-Shafer Theory to Map Representation

In the classical Bayesian approach of the occupancy grid, each cell contains a probability of being occupied, whereas Dempster-Shafer theory allows multiple masses to be stored in a single cell and offers formulas for updating these masses.

The basic idea of the DST (Dempster-Shafer Theory) is that uncertainty in knowledge can be represented as a mass function (a set of probabilities) that indicates what possible outcomes exist and how likely each of them is. This mass function can be used to estimate the degree of confidence in conclusions based on fuzzy and uncertain data.

The DST allows us to express ignorance, which means that some mass can be attributed to the set of all possible hypotheses. This ability to model uncertainty, including both aleatoric (inherent randomness) and epistemic (due to lack of knowledge) uncertainty, allows for better decision making in SLAM applications.

Also, DST is effective in combining evidence from multiple sources because it considers the reliability of each source when updating belief functions, which can be useful when using multiple sensors to estimate robot position and map.

In the Bayesian approach, conflicting evidence can lead to a weakening of the overall belief. Dempster-Shafer theory, on the other hand, allows contradictory evidence to be treated explicitly, resulting in a more nuanced representation of uncertainty.

Dempster-Shafer theory can in some cases be computationally more efficient than Bayesian methods because they do not require estimating the full joint probability distribution. Instead, they work with belief functions that can be updated using local computation. This makes them more suitable for real-time SLAM applications where computational resources may be limited and updates must be performed quickly.

## IV. Conclusion

In conclusion, solving the Simultaneous Localization and Mapping (SLAM) problem is crucial for autonomous robot navigation in unknown environments. This article presented various approaches and methods for addressing this challenge in both 2D and 3D cases, highlighting the unique advantages and limitations of each method. In particular, the application of Dempster-Shafer theory and Transferable Belief Model in SLAM algorithms, such as VinySLAM, has demonstrated potential for providing a more robust representation of cell states in an Occupancy Grid Map compared to traditional Bayesian probability approaches.

## References

[1] Huletski A., Kartashov D., Krinkin K., “Vinyslam: an indoor slam method for low-cost platforms based on the transferable belief model”, *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).*, 2017, pp. 6770-6776.

[2] Durrant-Whyte H., Bailey T. “Simultaneous localization and mapping: part I”, IEEE robotics & automation magazine., 2006, T. 13., №. 2., pp. 99-110.

[3] Grisetti G., Kümmerle R., Stachniss C., Burgard W. “A tutorial on graph-based SLAM.”, IEEE Intelligent Transportation Systems Magazine, 2010, pp. 31–43.
[4] Grisetti G., Tipaldi G. D., Stachniss C., Burgard W., Nardi D. “Fast and accurate SLAM with Rao–Blackwellized particle filters”, Robotics and Autonomous Systems, 2007, pp. 30-38.
[5] W. Hess, D. Kohler, H. Rapp, D. Andor, “Real-time loop closure in 2D LIDAR SLAM,” in Proc. 2016 IEEE Int. Conf. on Robotics and Automation, pp. 1271-1278, 2016.
[6] Siegwart R., Nourbakhsh I. R., Scaramuzza D., *Autonomous mobile robots*. A Bradford Book, 2011.
[7] B. Steux, O. E. Hamzaoui, “tinySLAM: A SLAM algorithm in less than 200 lines C-language program,” in Proc. 11th Int. Conf. on Control Automation Robotics Vision, pp. 1975-1979, 2010.
[8] P. Smets, R. Kennes, “The transferable belief model,” Artificial Intelligence, vol. 66, pp. 191-234, 1994.
[9] A. P. Dempster., "Upper and Lower Probabilities Induced by a Multivalued Mapping." Ann. Math. Statist. 38 (2), pp. 325 - 339, 1967