Building 3D Dense Reconstructions using LiDAR from a Walking Robot

Marcelo Gennari do Nascimento

Wadham College, University of Oxford

January 25, 2018

Acknowledgments

I like to aknowledge ...

Abstract

Abstract text goes here

Contents

1 Introduction

Autonomous robots are going to be one of the major achievements of science to the benefit of the public. Autonomy though depends on two main problems that are tightly related to each other: Localization and Mapping. The first concerns the problem of mapping the environment given a prior robot's trajectory. Most of the time though, neither the trajectory nor the map is known a priori.

Since the 1986 IEEE Robotics and Automation Conference, researchers have framed the general problem of Simultaneous Localization and Mapping (SLAM) as the "holy grail" of modern robotics [1]. A reliable solution to this problem would make autonomy one step closer to reality. Since the conference, a number of algorithms have been developed that successfully tackle SLAM, each of them with their advantages and drawbacks.

This project is concerned with putting together state of the art algorithms for SLAM and reconstruction systems to reliably and effiiciently build a 3D map of an environment with LiDAR data from a walking robot without any prior map or prior trajectory available. The success of this project would contribute to making the usage of mobile robots for tasks such as search and rescue and exploration one step closer to reality.

1.1 Aim of the Project

1.2 Organisation of the Report

Section 2 of the report will go through the literature review of the main building blocks of the project. In particular, we will explore the current developments in the solutions of the Simultaneous Localization and Mapping (SLAM), the Iterative Closest Points (ICP) and 3D Reconstruction problem.

Section 3 will go deeper on the system pipeline of the project. In order to get a general understanding of the system, detaied explanation of the steps of data processing, inputting and outputting will be explained. This will also introduce the two main parts of the system that will be explained in subsequent sections.

Section 4 explores the solution of the SLAM problem adopted in the project. Since building an accurate map is an essential step to the success of the project, this section will explain the details of how the SLAM solution works and why this particular structure was chosen.

Section 5 shows how the output from the SLAM solution will be integrated with the 3D reconstruction system.

Section 6 will conclude the report with an overall evaluation of how the system performed. It will also indicate ways in which the system could be improved in a subsequent project. An analysis of how the outcome of this project compares with similar systems will be provided.

2 Literature Review

2.1 Simultaneous Localization and Mapping (SLAM)

SLAM is the problem of whether it is possible for a mobile robot to create a globally consistent map of an environment and localize itself on it without prior knowledge of the map [1][2]. Building a map of an environment is a crucial step towards autonomy, since planning and control assume prior knowledge of mapping and localization. Mathematically, we can frame SLAM as a Markov Chain, a Bayes Net or a Factor Graph. Defining:

- $\mathbf{x_k}$: the pose of the robot (being $\mathbf{X_{0:k}}$ as the poses from time $\mathbf{0}$ to $\mathbf{k}$)

- $\mathbf{u_k}$: the odometry measurement ($\mathbf{U_{0:k}}$ as the historical measurements)

- $\mathbf{l_k}$: the landmark observation ($\mathbf{L_{0:k}}$ as the historical landmarks)

- $\mathbf{c_k}$: the loop closures

It is possible to formulate the problem of SLAM more formally using both a factor graph or a probabilistic framework:
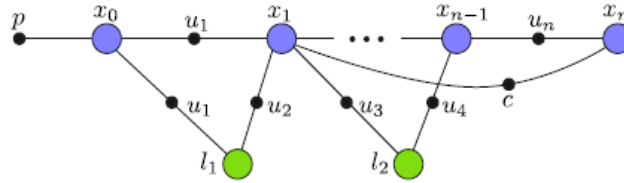
Figure 1: SLAM as a Factor Graph. The value $\mathbf{p}$ denotes a prior; $\mathbf{x_n}$ denotes the state vector; $\mathbf{u_n}$ denotes odometry measurements; $\mathbf{c}$ denotes close loops; $\mathbf{l_n}$ denotes landmark positions.

The graph shown in Figure 1 is a Factor Graph representation of the dependencies between variables and measurements. A probabilistic framework can then be extracted from it.

$$P(X, L, U, Z, C) \propto P(x_0) \prod_{i=1}^{M} P(x_i | x_{i-1}, u_i) \prod_{k=1}^{K} P(z_k | x_{i_k}, l_{j_k}) \prod_{w=1}^{W} P(x_i | x_{i_w}, c_w) \qquad (1)$$

The general goal of probabilistic SLAM is to find variables $\mathbf{X}^*$ and $\mathbf{L}^*$ that maximizes the posterior probability distribution of the Equation 1. Also, for the rest of this chapter, it will be assumed that the data association problem is solved and the indices i_w and i_k are known.

Theoretically, the mathematical analysis of SLAM has been shown that it is indeed feasible to build a nondivergent map with no prior information, so it is widely accepted that in theoretical grounds,

SLAM is a solved problem [1][2][3][4]. However, there are computational and algorithmic challenges that hinders the development of a real-time implementation system that performs SLAM. This problem gets even more complicated when considering unstructured environments and large scale maps [5].

There are three main implementations of SLAM: Kalman Filtering and Extended Kalman Filtering (with early works such as proposed by R. Smith [6]), Rao-Blackwellized Particle Filtering (most notably with FastSLAM [7]) and Information Filter (with the now state-of-the-art work of iSAM from Michael Kaess [8]).

2.1.1 Kalman Filtering

As one of the first implementations to appear to solve the problem of SLAM, the Extended Kalman Filtering (EKF) approach makes two assumptions when formulating the probabilistic SLAM: the process model (also referred to motion model in mobile robotics) and the observation model.

$$\textbf{Process Model}: \qquad\qquad\qquad \textbf{Observation Model}:$$

$$x_i = f(x_{i-1}, u_i) + w_i \qquad\qquad\qquad z_i = h(x_{i_k}, l_{j_k}) + + v_k$$

where w_i is white noise with covariance Q and v_k is white noise with covariance R. The process (motion) model is usually a generalization of how the robot moves (kinematics) [7][9]. As a consequence, the probability distributions $p(x_i|x_{i-1}, u_i)$ and $p(z_i|x_i, l_i)$ can be modeled as Gaussians with mean f and h and covariances Q and R. Modern analysis of the SLAM algorithms have shown that due to linearization of the functions f and h, assumptions about Gaussian noises and time-dependence of the process and observation model can cause the EKF solution to perform poorly unless many loop closures are detected in frequent intervals [10].

It is also known that the Kalman Filter approach requires storage of the order of $O(N^2)$ (where N is the number of features), and for a classic implementation of the algorithm, it also requires computational power of the order of $O(N^2)$ [3]. New methods for computing the covariances (which cause the squared dependence) by exploring state augmentation, partitioned updates and sparsity in the matrices have demonstrated faster solutions, thus requiring less computational power [5].

2.1.2 Particle Filtering

[7][9]

2.1.3   Information Filter

2.2   Iterative Closest Points (ICP)

2.3   3D Reconstruction Systems

3 System Pipeline

The pipeline for the overall system can be seen below. It consists of 4 modules: (I) First odometry estimates (based on either Visual Odometry or Wheel Odometry); (II) Laser Odometry (using Simona Nobili's AICP [11]); (III) Loop closure detection and Graph Optimization (using iSAM [8]); (IV) 3D dense volumetric reconstruction (using BOR^{2}G-CUBES [12][13]).

These 4 modules are broken down in two parts: the first part is concerned about solving the SLAM problem, which outputs a reliable trajectory and map. The second part is concerned about inputting that to a 3D reconstruction system (which in this case if BOR^{2}G-CUBES) to build the environment volumetrically.

4   SLAM Solution

As already introduced in Section 3, the SLAM solution has 4 independent building blocks. This offers flexibility enough to test the effect that each part has in the outcome of the map. This section is going to discuss deeper how the solution works with all the building blocks working together.

4.1   First Odometry Measurements

The first odometry measurements offer a relatively innacurate estimation of the pose of the robot at a certain time. It is used as the input pose to the AICP algorithm in order for it to process laser odometry based on the LiDAR scan.

During the project, two mehods to get the first odometry measurements were used. When getting data with the mobile robot Husky [14], the wheel odometry provided by the system served as a good first estimate of its position. When getting the data with the MultiSense SL [15], Visual Odometry from MultiSense's stereo camera was performed to get the first odometry estimations. A graph of both paths in world coordinates are shown below for comparison, along with the LiDAR scans viewed at bird's-eye view:

4.2   Laser Odometry

Given the first odometry measurements, the AICP algorithm was used to incorporate the LiDAR scan of each of the paths to the estimation problem. For the Husky example, the Velodyne [16] LiDAR was used, whereas for the Oxford Dataset, the MultiSense SL LiDAR was used.

The results for each of the paths are shown below for comparison. It is noticeable the difference between the accuracy of the First Odometry Measurements to the Laser Odometry. However, it can be seen that there is drift due to how the AICP algorithm works.

4.3   Loop Closure Detection

In order to adjust for the innacuracies of the path estimate, a loop closure detection algorithm was placed. Since it was assumed that the robot would not overdrift, three simple heuristics were used to detect loop closure: Time Filter Sampling, Euclidean Distance, and Pose Alignment.

- **Filter Sampling:** In order to avoid the problem of loop closing two state estimates that are in the same room but did not leave the room, a time filter sampling was applied. The idea is to sample every S state estimates, so that the candidate loop closing states would be sparse

enough so that the loop just occurs between big time frames, but dense enough to apply euclidean distance.

- **Euclidean Distance:** Once the estimates have been sampled, the euclidean distance between each of them is applied. If the distance between any two states is less than a threshold value E, then the two estimates form a candidate pair and they are added in the candidate pair list.

- **Pose Alignment:** After the resampling, the candidates pairs are tested for their relative pose. Since the loop closure correction relies on alignment of point clouds using ICP, it is a good strategy to select poses that would maximize the overlap between the point clouds associated with the candidates. In order to do that, another resampling is done based on the overlap between the field of view of the estimates. If the overlap is less than a threshold value O, then the candidate pair is dropped out from the list, and the remaining are the loop closures that are considered.

The parameters S, E and O are chosen by hand depending on what dataset is used in the algorithm. Changing the parameters can affect the outcome of the map significantly, so it is important to spend some time tuning them to the specific requirements of the test.

4.4   Graph Optimization
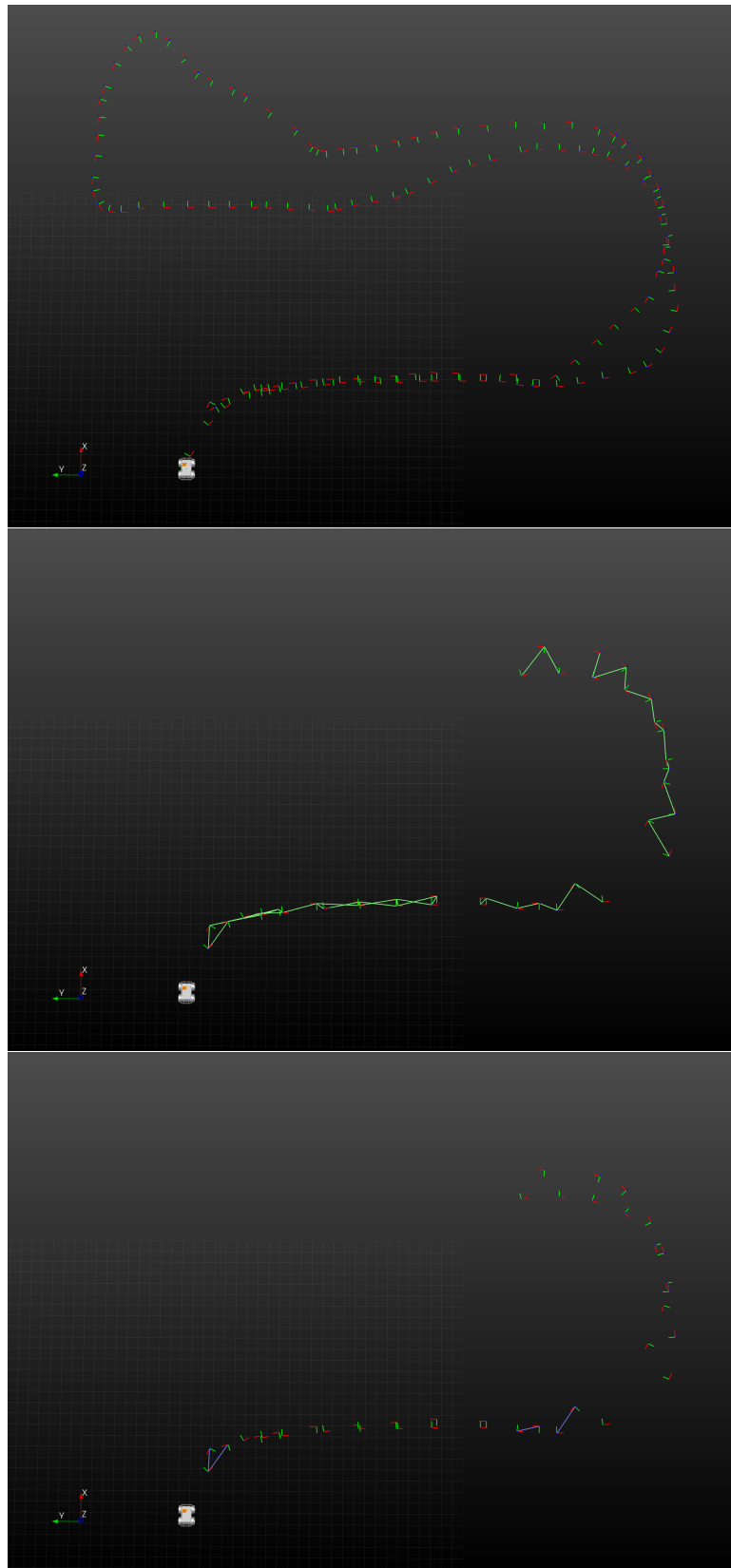
Figure 2: The three heuristics for loop closure detection

5 Integration with BOR^{2}G-CUBES

6   Conclusion

References

[1] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part i," *IEEE Robotics Automation Magazine*, vol. 13, pp. 99–110, June 2006.

[2] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. Leonard, "Past,present, and future of simultaneous localization and mapping: Towards the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[3] M. Csorba, "Simultaneous localisationa and map building," 1997.

[4] M. W. M. G. Dissanayake, P. Newman, S. Clark, 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, Jun 2001.

[5] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (slam): part ii," *IEEE Robotics Automation Magazine*, vol. 13, pp. 108–117, Sept 2006.

[6] R. Smith, M. Self, and P. Cheeseman, "Autonomous robot vehicles," ch. Estimating Uncertain Spatial Relationships in Robotics, pp. 167–193, New York, NY, USA: Springer-Verlag New York, Inc., 1990.

[7] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "Fastslam: A factored solution to the simultaneous localization and mapping problem," in *In Proceedings of the AAAI National Conference on Artificial Intelligence*, pp. 593–598, AAAI, 2002.

[8] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Incremental smoothing and mapping," *IEEE Trans. on Robotics (TRO)*, vol. 24, pp. 1365–1378, Dec. 2008.

[9] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte carlo localization for mobile robots," in *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*, vol. 2, pp. 1322–1328 vol.2, 1999.

[10] S. Huang and G. Dissanayake, "A critique of current developments in simultaneous localization and mapping," *International Journal of Advanced Robotic Systems*, vol. 13, no. 5, p. 1729881416669482, 2016.

[11] S. Nobili, R. Scona, M. Caravagna, and M. Fallon, "Overlap-based icp tuning for robust localization of a humanoid robot," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4721–4728, May 2017.

[12] M. Tanner, P. Piniés, L. M. Paz, and P. Newman, "BOR2G: Building Optimal Regularised Reconstructions with GPUs (in cubes)," in *International Conference on Field and Service Robotics (FSR)*, (Toronto, ON, Canada), June 2015.

[13] M. Tanner, P. Piniés, L. M. Paz, and P. Newman, "DENSER Cities: A System for Dense Efficient Reconstructions of Cities," *ArXiv e-prints*, Apr. 2016.

[14] Clearpath Robotics, "Husky a200," 2018.

[15] Carnegie Robotics, "Multisense sl," 2018.

[16] Velodyne LiDAR, "Velodyne," 2018.