2018 Latin American Robotic Symposium, 2018 Brazilian Symposium on Robotics (SBR) and 2018 Workshop on Robotics in Education (WRE)

Keyframe-based RGB-D SLAM for Mobile Robots with Visual Odometry in Indoor Environments using Graph Optimization

J. C. V. Soares and M. A. Meggiolaro
Robotics Laboratory - Department of Mechanical Engineering
Pontifical Catholic University of Rio de Janeiro (PUC-Rio)
Rio de Janeiro, Brazil
Email: virgolinosoares@gmail.com, meggi@puc-rio.br

***Abstract*—The SLAM problem is currently one of the most important topics in mobile robotics, due to the high number of applications that need its solution. This work proposes a methodology to perform SLAM in indoor environments with RGB-D data. The robot motion is estimated using FOVIS, a robust visual odometry system, and a graph-based probabilistic approach is used to minimize the errors caused by the drift in visual odometry. A keyframe selection approach is used to construct the graph and the g2o framework is used for the optimization. The proposed methodology is implemented as a Robot Operating System (ROS) package, and it is evaluated using benchmark datasets available in the literature. A comparison is made with state-of-the-art methods.**

***Keywords*-SLAM; RGB-D; graph optimization, FOVIS;**

I. Introduction

The simultaneous localization and mapping (SLAM) problem is one of the most researched topics in mobile robotics. It consists in creating a map of the environment and, concurrently, estimating the pose of the robot in a global coordinate system, assuming no prior knowledge about the environment. It has several applications, specially in indoor problems, with the abscence of GPS information [1].

For many years the laser range-finders were the most popular sensor for SLAM, due to its high range and precision. However, RGB-D sensors are an interesting alternative, due to a lower price and for the availability of color and depth information.

Usually the robot does not have prior information about the environment and has only sensor information, such as images, range measurements and odometry. However, sensor information is inherently urcertain and a probabilistic formulation is needed to reduce the errors caused by the accumulated drift in the motion estimation. There are three main probabilistic approaches for SLAM: kalman filters, particle filters and graph-based. First presented by Lu and Milios [2], the graph-based approach uses a graph to represent the trajectory of the robot, in which the poses are represented by the nodes, and measurement constraints between the poses are represented by the edges [3]. The graph is optimized using a least-squares formulation and the measurement errors are minimized.

The fundamental step that enables the graph optimization is the loop closure problem, which consists in detecting if the robot is in a previously visited region [4], allowing the robot to understand the real topology of the environment [1].

This work proposes a methodology that combines the information of a robust visual odometry system with a keyframe selection method to detect loop closures, and a state-of-the-art graph optimizer to solve the SLAM problem.

The implementation is made on the ROS framework [5]. ROS is built as a large number of small programs that communicate one another through messages, carried by topics. It works with a graph architeture, where each program is a node and the topics are the edges.

The paper is organized as follows. In section 2 is presented the related work. The section 3 details the proposed SLAM algorithm. Section 4 presents the results of the evaluation using real world datasets and a comparison between the proposed methodology and state-of-the-art methods. Finally, in section 5 is shown the final remarks.

II. Related Work

The use of RGB-D cameras for robotics research is relatively recent. Henry et al. [4] developed in 2012 a RGB-D SLAM system combining sparse visual features with Iterative Closest Point (ICP) to create and optimize a pose-graph, using the concept of keyframes [6] for graph construction and loop detection.

In 2011, Huang et al. [7] implemented a visual odometry system, named FOVIS, in a micro air vehicle with an RGB-D camera for localization and mapping using sparse visual features, but without a probabilistic formulation. Another visual odometry formulation for RGB-D sensors was developed by Whelan et al. [8] in 2013, integrating FOVIS and other visual odometry estimation methods with a GPU-based implementation, also without a probabilistic approach or loop closure detection.

In 2014, Silva and Gonçalves [9] developed a visual odometry and mapping system for RGB-D cameras in indoor environments using only CPU.

Another RGB-D SLAM system was developed by Endres et al. [10] in 2012, using sparse visual features and graph op-

978-1-5386-7761-2/18/$31.00 ©2018 IEEE
DOI 10.1109/LARS/SBR/WRE.2018.00026

timization. In 2014, Engel et al. [11] presented an algorithm called LSD-SLAM, also based on keyframes and pose-graph optimization, with, however, a monocular camera.

In 2017, Mur-Artal and Tardós [12] developed ORB-SLAM2, a complete SLAM system for monocular, stereo and also RGB-D cameras, using graph optimization. ORB-SLAM2 has good accuracy and performance. However, it creates a sparse representation of the environment. A dense map is achievable only with post-processing.

This work combines a keyframe loop detection approach and the robust visual odometry estimation of FOVIS to create a pose-graph. The graph is optimized using g^2o [13], a state-of-the-art framework for pose-graph optimization, to create a consistent trajectory and, consequently, an aligned dense point cloud global map.

III. RGB-D SLAM

A. Problem Definition

The graph approach for the SLAM problem can be subdivided into two main parts: front-end and back-end. The front-end is responsible to process sensor information and construct the graph. In the back-end the graph is optimized [3]. Fig. 1 shows the diagram of a Graph-SLAM system.

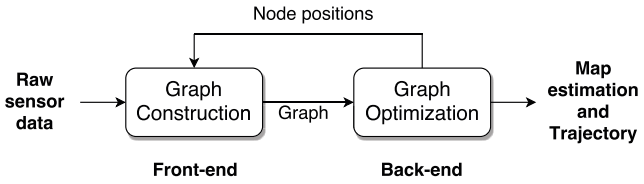

Figure 1. Graph-SLAM system

Every time a new pose is estimated with visual odometry, a node is added to the graph and an edge is created between the past and the current node, with their relative transformation. An edge is also created between two nodes if the system detects that the robot is in a previously visited location, what is called loop closure.

Once the graph is constructed, the objective of the graph-based SLAM methodology is to find the configuration of the poses that best explains these measurement constraints created by visual odometry and loop closures. Therefore, the objective is to find maximum belief of the state x, given the measurements z [1], as stated by Eq. (1).

$$\hat{x} = \underset{x}{\operatorname{argmax}}\, p(x|z) \tag{1}$$

Using Bayes' theorem, it becomes the likelihood of measurements given the state:

$$\hat{x} = \underset{x}{\operatorname{argmax}}\, \frac{p(z|x)p(x)}{p(z)} \tag{2}$$

Assuming no prior information and independent measurements, the problem factorizes into:

$$\hat{x} \propto \underset{x}{\operatorname{argmax}} \prod_{k=1}^{n} p(z_k|x) \tag{3}$$

Every measurement is assumed to be locally Gaussian:

$$\hat{x} = \underset{x}{\operatorname{argmax}} \prod exp(-e_{ij}^T(x_i, x_j, z_{ij})\Omega_{ij}e_{ij}(x_i, x_j, z_{ij})) \tag{4}$$

where Ω is the information matrix associated with each measurement, and e_{ij} is the error vector that states the difference between the measurement predicted by odometry and the actual measurement. Taking the logarithm to transform the product into a sum:

$$\hat{x} = argmin \sum_{ij} e_{ij}^T(x_i, x_j, z_{ij})\Omega_{ij}e_{ij}(x_i, x_j, z_{ij}) \tag{5}$$

Therefore, the graph-based probabilistic formulation is analogous to a non-linear least squares optimization problem [14].

B. Proposed SLAM Methodology

Fig. 2 shows the schematic overview of the proposed approach. The RGB-D data is used to create point clouds and to construct the graph. The graph is constructed using the poses estimated by FOVIS and the relative transformations between two poses, calculated in the loop detection process. The g^2o framework is used to store the nodes and edges of the graph and to perform the optimization. The final output of the system is the global point cloud map and the optimized trajectory of the robot.

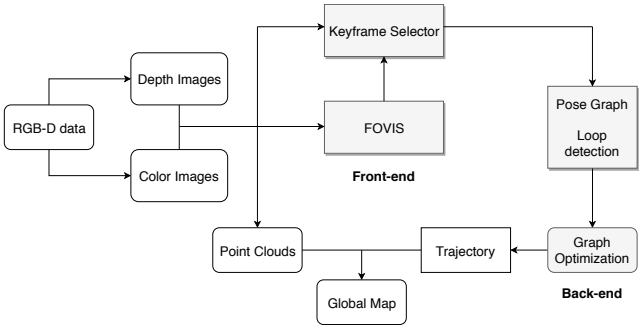

Figure 2. Schematic overview of the proposed approach

C. Data Acquisition

A point cloud is generated combining color and depth images provided by the RGB-D sensor, using the Eqs. (6) (7) and (8). These equations map a point (v, u) in the image plane to a 3D coordinate, and the variables c_x, c_y, f_x and f_y are intrinsic parameters of the RGB-D camera.

$$Z = depth\_image[v, u] \tag{6}$$

$$X = (u - c_x)\frac{Z}{f_x} \tag{7}$$

$$Y = (v - c_y)\frac{Z}{f_y} \tag{8}$$

Due to the large amount of memory required by the point cloud representation, a voxel grid filter is applied to reduce the number of points used to create the global map. Fig. 3 shows a point cloud with the full set of measured points and in Fig. 4 is shown the same point cloud after the filtering process.

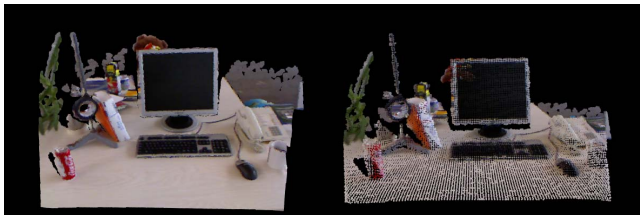

Figure 3. Point Cloud

Figure 4. Downsampled Cloud

D. Visual Odometry

The FOVIS library [7] is used to obtain a pose estimation. FOVIS uses the camera parameters, depth images and color images to estimate the current pose of the robot. Each pose i has the following format:

$$\mathbf{x}_i = [x_i, y_i, z_i, q_x, q_y, q_z, q_r] \tag{9}$$

where x_i, y_i, z_i are the euclidean coordinates of the robot and q_x, q_y, q_z, q_r compose the unit quaternion that represents the orientation of the robot.

E. Keyframe selection and Loop Closure

The incremental frame-to-frame alignment of FOVIS accumulates drift over time, due to sensor noise, which is the reason to use the Graph-SLAM formulation. The loop closure detection is crucial for graph optimization, as it defines the error constraints that will be optimized. Also, the loop detection allows the robot to understand the real topology of the environment. Otherwise it would see the world as an "infinite corridor".

Ideally, to detect a loop closure, it would simply require a comparison between the current frame and all past frames. However, it is computationally infeasible [4]. To overcome this problem, only a subset of frames is selected to be compared. They are called keyframes.

The first frame is selected as a keyframe and is matched against the next frames. The ORB features [15] are used in this methodology for the matching process. Besides their efficiency, they provide good invariance to changes in illumination and motion. Fig. 5 shows the matched points between two keyframes.

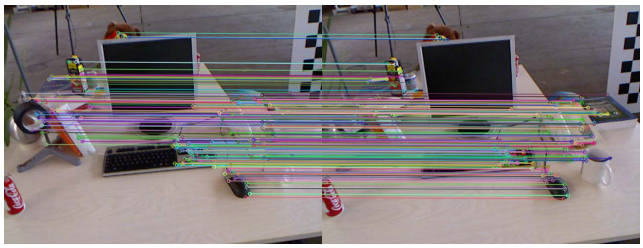

Figure 5. Feature Matching

The matching process is prone to wrong associations, called outliers. To overcome this problem, an outlier rejection filter is applied using the fundamental matrix. The fundamental matrix is the 3x3 matrix that relates two sets of matched points x and x' from two frames, as stated in Eq. (10). The matrix is computed using the RANSAC method [16], that detects the outliers and select the inliers, the correct associations, applying Eq. (10) for each correspondence, using a random sampling of the data. If the result is below a threshold close to zero, then the correspondence is an inlier.

$$x_i' F x_i = 0,\ i = 1 \ldots n \tag{10}$$

When the number of matched inliers between two frames is below a threshold, it means the robot has made a significant movement and a new keyframe is chosen. Every new keyframe is matched againt the previous ones. However, two keyframes are only compared if their global pose is close enough, given a threshold.

Every time a new keyframe is detected, a node is added to the graph with the corresponding position and orientation. If the number of matched inliers between the current keyframe and a past keyframe is above a threshold, called loop closure inliers threshold, then a loop is detected and an edge is added to the graph. This edge is composed by the transformation between the two frames and the associated information matrix, that comprises the uncertainty of the measurement. The information matrix is created based on the number of inlier features detected in the loop.

The ICP algorithm is used to determine the relative transformation between two keyframes with near global pose, using their corresponding point clouds. The following algorithm details the procedure of keyframe selection and loop closure detection in pseudocode notation.

1: **for** every new frame i **do**
2: $P_i \leftarrow i^{th}$ fovis pose
3: detect features
4: feature match(features i, features i-1)
5: $n\_inliers \leftarrow$ outlier rejection
6: **if** $n\_inliers < threshold$ **then**

```
    keyframe_j ← currentframe_i
    graph vertex j ← P_j
    compute T_{j-1,j} between keyframes (FOVIS)
    add edge
    compare current and past keyframes
    if loop detected with keyframe k then
      compute T_{k,j} between keyframes (ICP)
      add edge
    end if
    endif
  end if
  endif
end for
endfor
```

F. Graph Optimization

The g^2o framework is used to register the graph and perform the optimization when a loop is detected. Every time a keyframe is selected, a node is created with the format "ID $x\ y\ z\ q_x\ q_y\ q_z\ q_w$", corresponding to the pose number and the respective 3D position and orientation in unit quaternion representation. When a loop is detected, an edge is created with the format "IDfrom IDto $x\ y\ z\ q_x\ q_y\ q_z\ q_w$ I11 ... I66", corresponding to the numbers of observing and observed nodes i and j, the translation vector between them and the unit quaternion rotation. The numbers "I11 ... I66" are the 21 top triangular elements of the 6x6 information matrix. The graph is optimized with the Levenberg-Marquardt algorithm.

G. Implementation Details

The system is implemented as a C++ ROS package with auxiliary header files. The implementations from the OpenCV library [17] are used for the feature detection and matching. The Point Cloud Library (PCL) [18] is used to store, visualize and manipulate the point clouds.

IV. RESULTS

This implementation is numerically evaluated using the TUM benchmark dataset [19], from the Technical University of Munich, consisting of several sequences of RGB and depth images, with their corresponding ground-truth trajectories. The two chosen sequences are effective to evaluate the hability of the system to detect loop closures, according to Sturm et al. [19].

All tests were conducted in a notebook with an Intel Core i7 6700 HQ processor with 2.60 GHz and 16 GB of RAM, running Ubuntu Linux 14.04 LTS and ROS Indigo.

A. Evaluation Metrics

The Absolute Trajectory Error (ATE) metric is used for the numerical evaluation. It consists in a comparison between the absolute distances of the estimated and ground-truth trajectories, evaluating the global consistency [19].

For each dataset is evaluated the mean, minimum, maximum and root mean square errors. Given the trajectory estimate with translational components $\hat{x} = [\hat{x}_1, \cdots, \hat{x}_n]$, and the ground truth trajectory with translational components $x = [x_1, \cdots, x_n]$, the root mean square error (RMSE) is given by Eq. (11).

$$RMSE = \left(\frac{1}{n} \sum_{i=1}^{n} \|\hat{x}_i - x_i\|^2 \right)^{1/2} \tag{11}$$

The ground truth trajectories have the format "timestamp tx ty tz qx qy qz qw", where timestamp is the time of each pose in unix epoch time, "tx, ty, tz" is the translation vector, and "qx qy qz qw" is a unit quaternion. The estimated trajectories and the ground truth trajectories are aligned using the timestamps of each pose [19].

B. Dataset Evaluation

The first sequence is called "fr1 room" and corresponds to a movement of $15.989m$ of ground-truth trajectory length, $0.334m/s$ of average translational velocity and $29.882deg/s$ of average angular velocity. Fig. 6 shows the comparison between the estimated trajectory and the ground-truth trajectory. Fig. 7 shows the resulted global point cloud map, and in Tab. I is shown the comparison between the ATE error with and without graph optimization.

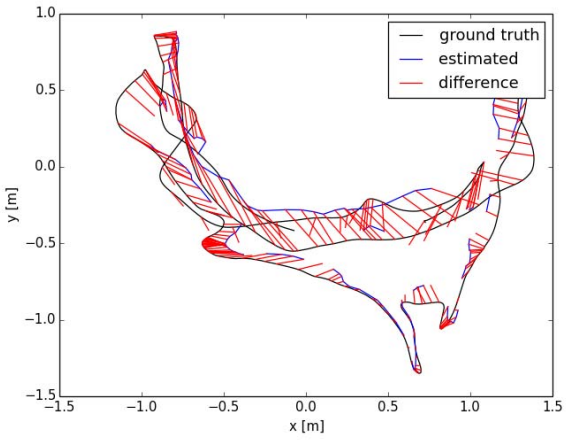

Figure 6. fr1-room - Optimized

Table I
ATE EVALUATION OF THE FR1 ROOM DATASET

Error (m)	FOVIS	This work
RMSE	0.2807	0.1987
Mean	0.2432	0.1722
Min	0.0256	0.0159
Max	0.6446	0.4072

The second evaluation is the "fr3 long office household" sequence, with $21.455m$ of ground-truth trajectory length,

Figure 7. fr1-room - Point Cloud Map

$0.249m/s$ of average translational velocity and $10.188deg/s$ of average angular velocity. Fig. 8 shows the comparison between the estimated trajectory and the ground-truth trajectory. In Figs. 9 and 10 is shown, respectively, the initial point cloud and the resulted global point cloud map. In Tab. II is shown the comparison between the ATE error with and without graph optimization.

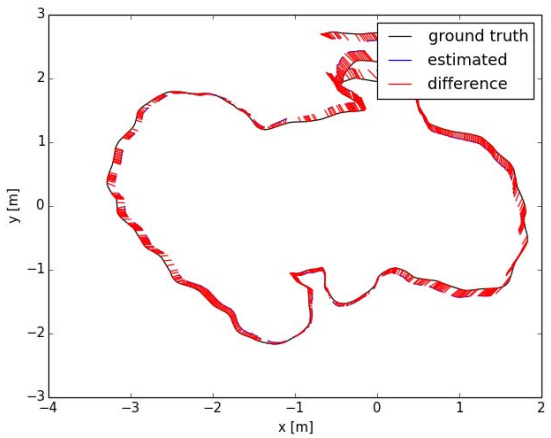

Figure 8. fr3-long-office - Optimized

Table II
ATE EVALUATION OF THE FR3 LONG OFFICE DATASET

Error (m)	FOVIS	This work
RMSE	0.2717	0.1238
Mean	0.2329	0.1101
Min	0.0381	0.0156
Max	0.5375	0.2685

Both results showed that the system was able to produce consistent maps of indoor environments, and the loop

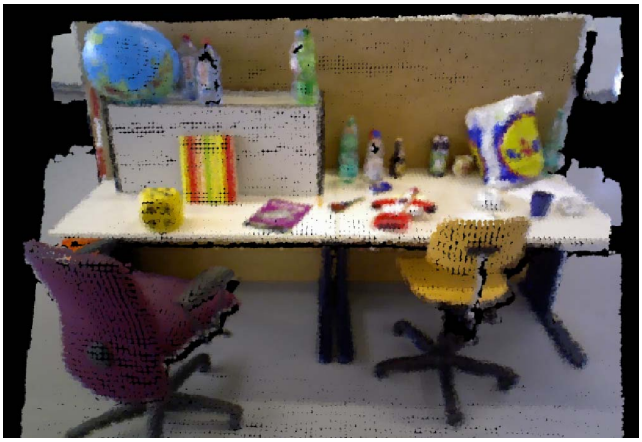

Figure 9. fr3-long-office: Initial Point Cloud

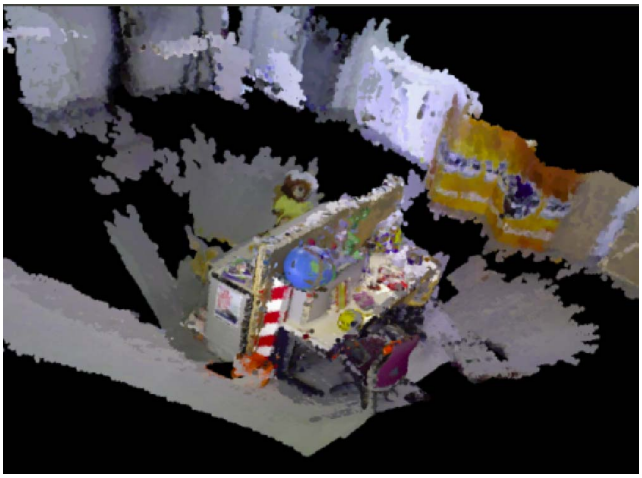

Figure 10. fr3-long-office: Point Cloud Map

closure and graph optimization provided a considerable improvement in the FOVIS trajectory estimation. The root mean square, maximum, minimum and mean errors were all minimized. The errors in the first evaluation are larger than the second one due to the higher translational and angular velocities of the camera.

In Tables III and IV is shown comparisons between the proposed implementation and the aforementioned methods from the literature. The comparisons are made using the ATE RMSE metric and the datasets freiburg room, with 3Hz of frame rate, and freiburg long office household, with 30 Hz of frame rate. For the freiburg room dataset, the comparison, shown in Tab. III, is made with the first version of the rgbdslam method, developed by Endres et al. [10] and FOVIS [7]. This work outperformmed both methods.

For the freiburg long office household dataset, the comparison, shown in Tab. IV, is made with LSD-SLAM [11] and FOVIS. This work showed satisfatory results for a real time performance, achieving lower errors than both methods.

Table III
COMPARATIVE RESULTS FOR THE FR1 ROOM DATASET

Method	ATE RMSE (m)
This work	0.1987
RGB-D SLAM	0.2190
FOVIS	0.2807

Table IV
COMPARATIVE RESULTS FOR THE FR3 LONG OFFICE DATASET

Method	ATE RMSE (m)
This work	0.3064
LSD-SLAM	0.3853
FOVIS	0.5144

V. CONCLUSION

In this paper an RGB-D SLAM algorithm for mobile robots in indoor environments was proposed. A visual odometry system combined with a keyframe loop closure approach was used to create a pose-graph, further optimized by a state-of-the-art framework. The dataset evaluation proved that the system is robust to solve the SLAM problem, reducing odometry errors and generating cohesive 3D maps. Furthermore, the implementation was compared with other methods from the literature, outperforming their results. Future improvements include the use of new methods to increase the performance of the loop closure system.

ACKNOWLEDGMENT

The authors of this work would like to thank CAPES and CNPq for the financial support.

REFERENCES

[1] 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, 2016, vol. 32, pp. 1309-1332.

[2] F. Lu and E. Milios, "Globally consistent range scan alignment for environment mapping", Autonomous Robots, 1997, vol. 4, pp. 333–349.

[3] G. Grisetti, R. Kümmerle, C. Stachniss and W. Burgard, A Tutorial on Graph-Based SLAM, IEEE Intelligent Transportation Systems Magazine, 2010, vol.2, pp. 31-43.

[4] P. Henry, M. Krainin, E. Herbst, X. Ren and D. Fox, "RGB-D mapping: using kinect-style depth cameras for dense 3D modeling of indoor environments", The International Journal of Robotics Research, 2012, vol.31, pp. 647-663.

[5] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler and A. Ng, "ROS: an open-source robot operating system", in Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA), Workshop on Open Source Robotics, 2009.

[6] K. Konolige and M. Agrawal, "FrameSLAM: From bundle adjustment to real-time visual mapping", IEEE Transactions on Robotics, vol.24, n.5, pp. 1066-1077, 2008.

[7] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox and N. Roy, "Visual odometry and mapping for autonomous flight using an RGB-D camera", in Int. Symposium on Robotics Research (ISRR), 2011.

[8] T. Whelan, H. Johannsson, M. Kaess, J. J. Leonard and J. McDonald, "Robust real-time visual odometry for dense RGB-D mapping", in Proceedings of IEEE International Conference on Robotics and Automation (ICRA), 2013, pp. 5724-5731.

[9] B. Silva and L. Gonçalves, "Visual odometry and mapping for indoor environments using RGB-D cameras", in Proc. of the IEEE Latin American Robotics Symposium, 2014.

[10] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, and W. Burgard, "An evaluation of the RGB-D SLAM system", in Proceedings of IEEE International Conference on Robotics and Automation (ICRA), 2012, pp. 1691–1696.

[11] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM", in Proceedings of Computer Vision – ECCV 2014: 13th European Conference, pp. 834–849.

[12] R. Mur-Artal and J. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras". IEEE Transactions on Robotics, v. 33, n. 5, p. 1255-1262, 2017.

[13] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige and W. Burgard, "g2o: A general framework for graph optimization", in Proceedings of IEEE International Conference on Robotics and Automation (ICRA), 2011.

[14] J. Soares, and M. Meggiolaro, "A pose-graph optimization tool for MATLAB", X Congresso Nacional de Engenharia Mecânica (CONEM), 2018.

[15] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF", in Proceedings of IEEE international conference on Computer Vision (ICCV), 2011, pp.2564–2571.

[16] M. Fischler and R. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography", Commum ACM, 1981, vol. 24, pp. 381 - 395.

[17] Itseez, "Open source computer vision library", https://github.com/itseez/opencv, 2015.

[18] R. Rusu and S. Cousins, "3D is here: point cloud library (PCL)", in Proceedings of IEEE International Conference on Robotics and Automation (ICRA), 2011.

[19] J. Sturm, N. Engelhard, F. Endres, W. Burgard and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems", in 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2012.