ExplORB-SLAM: Active Visual SLAM Exploiting the Pose-graph Topology

Julio A. Placed$^{(\boxtimes)}$, Juan J. Gómez Rodríguez, Juan D. Tardós,
and José A. Castellanos

University of Zaragoza, 50018 Zaragoza, Spain
{jplaced,jjgomez,tardos,jacaste}@unizar.es

Abstract. Deploying autonomous robots capable of exploring unknown environments has long been a topic of great relevance to the robotics community. In this work, we take a further step in that direction by presenting an open-source active visual SLAM framework that leverages the accuracy of a state-of-the-art graph-SLAM system and takes advantage of the fast utility computation that exploiting the structure of the underlying pose-graph offers. We achieve fast decision making through careful estimation of *a posteriori* weighted pose-graphs and by employing a utility function that balances exploration and exploitation principles.

Keywords: Active SLAM · Graph topology · Autonomous exploration

1 Introduction

A CTIVE simultaneous localization and mapping (SLAM) names the problem of autonomously deciding where a robot should move next in order to form the most accurate representation of the environment (i.e., including localization and mapping) possible. Thus, it refers to the problem of decision-making on control commands for a robot which is performing SLAM and that is capable of navigating in an unknown environment. See [1] and the references therein.

Traditional approaches to solve active SLAM born within the robotics community are numerous and vary in shape and form [2–5]. Most of them share, however, the following division in three discernible stages [1,6]:

- Goal identification: to reduce the (ideally infinite) number of possible destinations, it is common to define a finite and typically low-dimensional subset of boundaries between the known and unknown regions, i.e., frontiers.
- Utility computation: the expected usefulness of executing a candidate set of actions can be estimated studying the expected uncertainty in the two target random variables (i.e., the robot location and the map); either leveraging information theory (IT) or theory of optimal experimental design (TOED).
- Selection and execution of the optimal action. After estimating utility, it all comes down to selecting the most useful destination and traveling to it.

The interest of this paper particularly lies on TOED-based approaches, that directly quantify utility from the variance of the variables of interest, under the assumption of Gaussianity. Feder *et al.* [6] were the first to use task-oriented metrics (i.e., *optimality criteria*) for active SLAM and, since then, their popularity has steadily grown due to their strong mathematical basis and elegant formulation. Their complexity and high computation burden, though, make information theoretic approaches still prevail in the literature [3,7].

In these active SLAM approaches, the second stage represents the main bottleneck as it requires analyzing the spectrum of the Fisher information matrices (FIMs) expected for a candidate set of actions. Such matrices must comprise the uncertainties of the robot state (e.g., pose) and the environment representation (i.e., map). For landmark-based maps, utility can be obtained either by studying a full (high-dimensional) FIM that contains both robot poses and landmarks [6], or by computing the uncertainty of the robot subject to the set of observations, i.e., marginalizing the landmarks' information [8]. In contrast to the above, Wang and Englot [9] incorporate the robot's expected variance into the landmarks' uncertainty. On the other hand, for dense metric maps (e.g., occupancy grids), most works resort to IT metrics, e.g., the entropy [3]. An exception is the work from Carrillo *et al.* [5], where a TOED-based metric of the uncertainty in the robot location is used to correct the entropy of the map. The opposite idea is employed in [10], where the authors heuristically embed the impact of the map's entropy on the location uncertainty metric.

For the case of active graph-SLAM, significant reductions in the time required to execute the second stage were recently achieved leveraging the insight of utility being closely linked to the topology of the underlying *pose-graph* [11,12]. Intuitively, the connectivity (i.e., the Laplacian spectrum) of the pose-graph can be regarded as a measure of how well the environment model is being formed. Later works [10,13] demonstrate the tight relationship between modern optimality criteria and connectivity indices, when the edges of the graph are weighted appropriately. Hence, the optimal actions to execute can be found by studying the topology of the expected weighted pose-graphs rather than analyzing the full posterior FIM. Successful applications that leverage the above include, for example, coverage tasks with uncertainty awareness [14] or 2D active SLAM [13].

1.1 Contributions

The main contribution of this work is an active visual SLAM system that exploits the topology of the underlying pose-graph to identify D-optimal candidate locations over affordable time horizons. From the accurate sparse map and trajectory estimation ORB-SLAM2 [15] provides, we design a decision-making mechanism that switches between *exploration* and *exploitation* principles. To do so, we adapt the insight of discounting information in Shannon-Rényi entropy [5] to operate in the task space, and leverage the theoretical results on pose-graphs' optimality from [13]. The above, along the method to predict reobservations and their associated information, emerge as side contributions of our work. The last contribution is the code release in the ExplORB-SLAM repository for ROS

Noetic, representing a complete framework to test this and other active SLAM approaches and aiming to pave the way for reproducible research in this field.

1.2   Paper Structure

The rest of the paper is organized as follows. Section 2 contains a brief introduction to visual graph-SLAM, modern optimality criteria, and their relationship with graph connectivity indices. In Sect. 3, we present and thoroughly describe the proposed system. Section 4 shows experimental validation results. Finally, the manuscript concludes in Sect. 5, where future work is also outlined.

2   Background

2.1   Visual Graph-SLAM

Visual SLAM refers to the problem of reconstructing the environment and locating an agent using only images from a set of cameras as inputs. This modality is a hot topic in both industry and academia as cameras are cheap sensors that also provide useful information for other computer vision tasks like object detection.

ORB-SLAM2 [15] is one of the state-of-the-art algorithms in V-SLAM and is the basic building block for this paper. It leverages the use of ORB features inside an accurate bundle adjustment (BA) optimization framework that minimizes the reprojection error of the estimated landmarks in order to refine its 3D position and the pose of the agent. In order to constrain the execution time, ORB-SLAM only performs BA at a lower rate than the video frequency using a subset of the images received, so called *keyframes*, that carry high visual innovation with respect the rest of the map.

An interesting property of BA is its sparsity pattern. With a simple study of its structure and its Hessian layout, one can realize that it has a sparse block-structure that can be also regarded as a graph that connects keyframes to the observed landmarks. To enhance computation efficiency of the algorithm, the landmarks can be marginalized out using the Schur complement obtaining the reduced camera system, whose Hessian only relates keyframes that have common observations. The corresponding graph is also simplified, forming the so called *pose-graph* in which vertices only represent keyframes and edges the relative pose between pairs of connected keyframes. ORB-SLAM, however, only builds this simplified graph when it needs to correct a loop. To reduce the computation burden, it uses an even sparser version of the pose-graph, the *essential graph*. The peculiarity of this graph resides in the fact that it only connects keyframes when they share a minimum amount of observations. Thus, keyframes with few landmarks in common are not connected, sparsifying the graph.

2.2   On the Optimality of SLAM Pose-graphs

During the second step in active SLAM, quantifying uncertainty is of utmost importance. On the basis of TOED, Kiefer [16] proves the existence of a family

of scalar mappings $\|\cdot\| : \mathbb{R}^{\ell\times\ell} \mapsto \mathbb{R}$ dependent of just one parameter p and which can be expressed in terms of the eigenvalues (λ) of the matrix to be quantified, e.g., covariance ($\boldsymbol{\Sigma}$) or FIM ($\boldsymbol{\Phi}$):

$$\|\boldsymbol{\Sigma}\|_p \triangleq \left(\frac{1}{\ell}\text{trace}(\boldsymbol{\Sigma}^p)\right)^{\frac{1}{p}} \tag{1}$$

Particularizing the above equation for the boundary values of p, modern *optimality criteria* result. The most relevant metric among them is D-optimality ($p = 0$), since it captures the global uncertainty [16]. On the downside, it also entails the highest computational complexity. Its modern formulation is as follows:

$$D\text{-}opt\,(\boldsymbol{\Sigma}) \equiv \|\boldsymbol{\Sigma}\|_0 \triangleq \exp\left(\frac{1}{\ell}\sum_{k=1}^{\ell}\log(\lambda_k)\right) \tag{2}$$

Let us now consider the specific (and somehow prevalent) case in which a graph-SLAM algorithm is used during active SLAM. These methods use a graph representation in which nodes encode the robot and landmark states, and edges encode the constraints between them (e.g., observations, odometry or loop closures). A flatter representation can be achieved by marginalizing the observations and thus only representing the robot states in nodes, i.e. a pose-graph $\boldsymbol{\mathcal{G}}$. In such case, the i-th vertex $\boldsymbol{v}_i \in \mathcal{V}$ will contain the robot pose (in the world frame) $\boldsymbol{T}_{wi} \in SE(n)$. On the other hand, an arbitrary edge $\boldsymbol{e}_j \in \mathcal{E}$ that connects the pair of vertices $\{\boldsymbol{v}_i, \boldsymbol{v}_k\} \in \mathcal{V}$ will encode the relative transformation between such pair, $\boldsymbol{T}_{ik} \in SE(n)$, and its associated uncertainty. Due to its higher sparsity, the latter is preferably represented by the FIM $\boldsymbol{\Phi}_j \in \mathbb{R}^{\ell\times\ell}$; with ℓ the degrees of freedom of the n-dimensional Euclidean space.

Decision-making in active SLAM can be reduced to evaluating optimality criteria over the expected FIM of the entire system at candidate goal locations, in order to compare them and later travel to the most informative one. This full FIM $\boldsymbol{Y} \in \mathbb{R}^{|\mathcal{V}|\ell\times|\mathcal{V}|\ell}$ must comprise the uncertainty of both the map and the robot state, as discussed in Sect. 1, and is commonly built upon the relative FIMs $\boldsymbol{\Phi}_j$. See [13] for further details on its structure. High dimensionality of $\boldsymbol{Y}$ makes the analysis of its spectrum computationally intensive and even intractable for online systems with large state and/or action spaces. By exploiting the facts that (i) the sparsity pattern of $\boldsymbol{Y}$ conveys that of the Laplacian of the underlying pose-graph and (ii) their spectra are intimately linked under certain conditions, optimality criteria of $\boldsymbol{Y}$ can be approximated by that of the (weighted) Laplacian of the graph. For the case of D-optimality, the following relationship may be established for the general case of graph-SLAM formulated over the Lie group $SE(n)$ [13]:

$$D\text{-}opt(\boldsymbol{Y}) \approx D\text{-}opt(\boldsymbol{L}_\gamma) = (|\mathcal{V}|\ t(\boldsymbol{\mathcal{G}}_\gamma))^{\frac{1}{|\mathcal{V}|}} \tag{3}$$

where $\boldsymbol{\mathcal{G}}_\gamma$ is the posterior pose-graph in which each edge is weighted with the scalar $\gamma_j = \|\boldsymbol{\Phi}_j\|_0$, $\boldsymbol{L}_\gamma$ its Laplacian matrix, $t(\boldsymbol{\mathcal{G}}_\gamma)$ its weighted number of spanning trees and $|\mathcal{V}|$ the dimension of the set $\mathcal{V}$ (i.e., the total number of vertices).
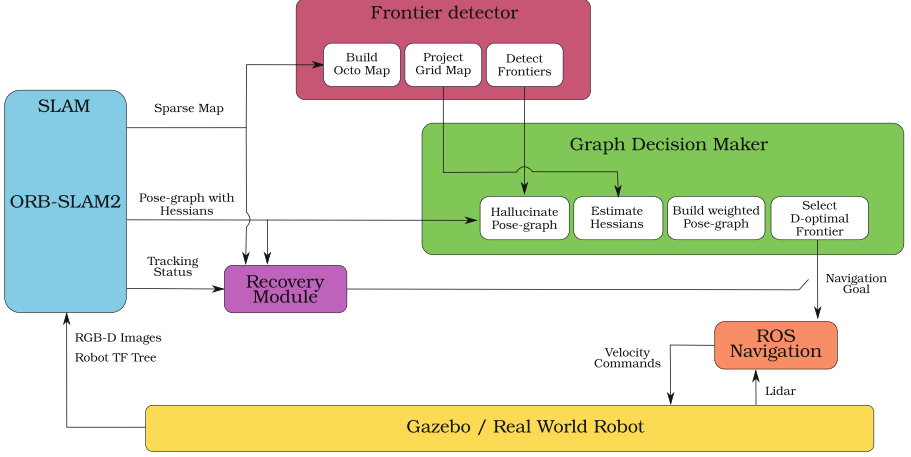
Fig. 1. Overview of ExplORB-SLAM.

3   Proposed Method

The proposed system leverages the fast computation of D-optimality using (3) to achieve online autonomous exploration while performing visual graph-SLAM. It consists of several different modules, which are depicted in Fig. 1 and described hereafter. The whole system is built within a ROS Noetic framework that eases the communication between modules and allows the connection with Gazebo.

3.1   SLAM

This fundamental module contains a modified version of ORB-SLAM2 [15] with a wrapper for ROS Noetic. In order to later perform decision-making, some changes were required. The most important one involves extracting the pose-graph and the information matrices[1] associated to the relative movement between vertices (i.e., keyframes) so as to build the required weighted pose-graph.

In a separate thread, we construct the camera-point Hessian of the SLAM system using the Gauss-Newton approximation to the least squares problem, i.e., the Jacobians of the reprojection error as in a BA [17]. It will have the form:

$$H'_{\text{SLAM}} = \begin{pmatrix} H_c & H_{cp} \\ H_{cp}^T & H_p \end{pmatrix} \tag{4}$$

where H_c and H_p are the blocks which define the information about the robot poses and the map points, respectively; and H_{cp} the correlation between them. Since we aim at reasoning over pose-graphs, map points must be marginalized. The reduced Hessian matrix can be computed using the Schur complement as:

---

[1] Throughout this paper, we will equally use FIM and Hessian matrix, since they are equivalent when evaluating the latter at the *maximum likelihood estimate.*
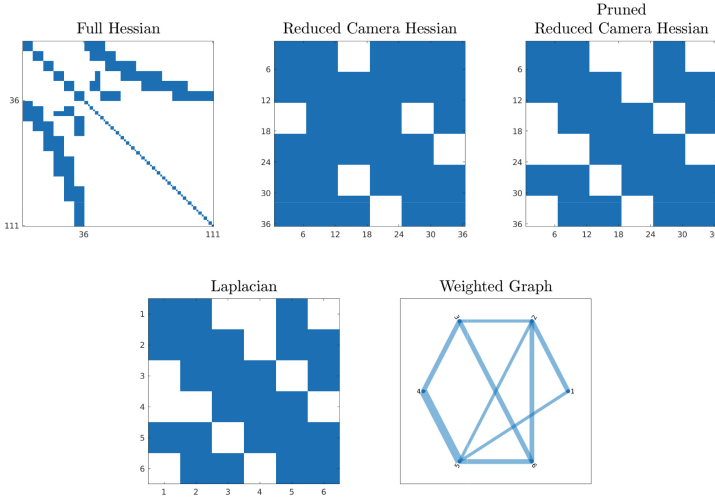
Fig. 2. Sparsity patterns of the Hessian and Laplacian matrices in a toy example with 6 poses and 25 map points. From top left to bottom right: SLAM Hessian $\boldsymbol{H}'_{\text{SLAM}}$, reduced Hessian before and after pruning connections with less than 3 observations in common $\boldsymbol{H}'_c$, Laplacian and resulting weighted pose-graph.

$$\boldsymbol{H}'_c = \boldsymbol{H}_c - \boldsymbol{H}_{cp}\ \boldsymbol{H}_p^{-1}\ \boldsymbol{H}_{cp}^T \tag{5}$$

The block-sparsity pattern of the above matrix is sufficient to define the pose-graph topology. The edge weights (dependent of the Hessian of the relative transformations) will be given by off-diagonal terms of $\boldsymbol{H}'_c$. To sparsify the pose-graph, we only keep those edges whose connected vertices share a minimum number of observations as in the essential graph, i.e., the intersection of the sets of observed landmarks is above a threshold. Figure 2 contains an example of the sparsity patterns of the Hessians and the Laplacian, and the resulting graph.

3.2   Frontier Detection

From the sparse 3D point cloud generated by ORB-SLAM, we build an Octomap and project it to the ground creating a 2D grid map in which frontiers can be spotted after several morphology transformations. Two frontiers detectors are used over the occupancy map, somehow following the popular work by Umari and Mukhopadhyay [4]. The first of them is based of rapidly-exploring random trees, while the second one employs the Canny edge detection algorithm. The candidates are clustered using the mean-shift algorithm, and then filtered to prune old points, low-informative returns and unreachable locations. The main limitation of this module is that the candidate search is restricted to the ground surface, limiting the algorithm application to wheeled robots. In future work, we plan to extend this module to detect 3D goals from sensor measurements, leveraging the insight that frontiers are bound to appear in recently sensed areas [18].
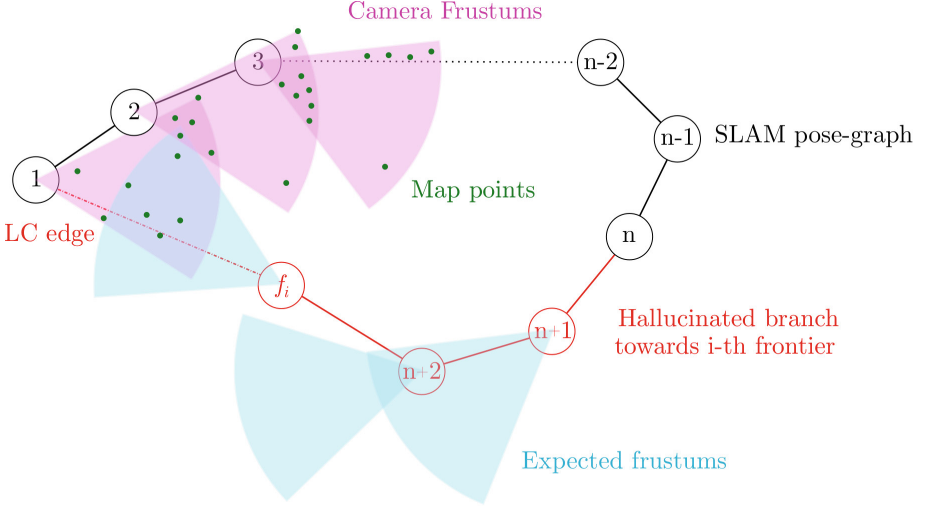
Fig. 3. Example of the graph hallucination process towards frontier f_i, considering $n_{p,min} = 3$ and $n_{p,max} = 6$. A loop closure edge with probability $p^{LC} = 1$ has been created between vertices "f_i" and "1".

3.3   Weighted Graph Hallucination

Graph Prediction. The pose-graph from ORB-SLAM is hallucinated towards every frontier spotted, thus creating as many graphs as frontiers. For each of them, we add a number of vertices along the expected path to reach the associated frontier; which is computed using Dijkstra's global planner from ROS navigation stack. The more complex or longer the path, the greater number of vertices will be in the hallucinated branch. We are aware that this method implies evaluating one single hypothesis for each frontier, although it has not been considered a critical issue. Initially, every vertex in the branch is just connected sequentially with its predecessor with an odometry constraint. After that, we consider the possibility of loop closures to appear. To do so, we first identify the set of existing map points expected to be observed from every hallucinated node (i.e., lie inside its expected frustum). If the number of covisible points with any other existing node in the SLAM graph, n_p, is higher than a certain threshold, $n_{p,min}$, the two will be connected by a loop closure edge. This edge will have the following probability of occurrence associated with it:

$$p^{LC} = \begin{cases} 0 & \text{if} \quad n_p < n_{p,min} \\ 1 & \text{if} \quad n_p > n_{p,max} \\ \frac{n_p}{n_{p,max}} & \text{otherwise} \end{cases} \qquad (6)$$

where $n_{p,max}$ is a defined upper bound. Figure 3 illustrates the process of graph prediction with a toy example.

Hessian Estimation. Once the topology of the hallucinated graph is defined, the FIMs associated to each edge in the hallucinated branch must be computed in order to later weight the graph. Assuming that visual odometry uncertainty will remain similar unless a loop closing occurs, the relative Hessians of odometry edges can be considered equal to the last one from the known SLAM pose-graph:

$$\boldsymbol{H}^{odom} = \boldsymbol{H}^{prev} \tag{7}$$

In the case that a loop closure is expected to occur between a pair of vertices, the Hessian will be given instead by the Jacobian matrices of the reprojection error of all covisible points and the likelihood of closing the loop, as shown in (6):

$$\boldsymbol{H}^{LC} = p^{LC} \sum_{i}^{n_p} \boldsymbol{J}_i^T \boldsymbol{J}_i \tag{8}$$

Hence, far and scarce covisible points will result in higher expected uncertainties.

The previous Hessians, however, do not account for the decrease in the environment's uncertainty as new regions are explored, since the complete set of landmarks is unknown. To also include it and motivate a balance between exploration and exploitation, we leverage the insight of discounting information from Carrillo *et al.* [5]. We adapt this concept to operate in the task space (i.e., uncertainty) rather than in the information space (i.e., entropy). Besides, instead of discounting from the map's entropy the information value of visiting a region with high localization uncertainty as in [5]; our approach builds upon the idea of penalizing the above Hessians if no new areas are visited. Just like the Shannon-Rényi entropy, the Hessian of the j-th edge will be given by:

$$\boldsymbol{H}_j = \boldsymbol{H} - \left(\frac{1}{1 - \alpha} \right) \boldsymbol{H} \tag{9}$$

where $\boldsymbol{H}$ is to be computed using (7) or (8) depending on the constraint type, $\alpha = 1 + \frac{1}{\sigma}$ and σ is a parameter that encodes the novelty of the regions to visit. More specifically, it is computed as the percentage of unseen area expected to be observed in the occupancy grid map within a 1.5 meter radius around the node.

Graph Weighting. Finally, every edge in the hallucinated pose-graph is weighted with the *D-opt* of the corresponding Hessian, i.e., $\gamma_j = \|\boldsymbol{H}_j\|_0$; so as to allow the use of (3).

3.4   Frontier Selection And Navigation

In order to identify the optimal frontier, we compute *D-opt* for each hallucinated graph using (3) and select that with the highest value:

$$f^* = \arg\max_f D\text{-}opt(\boldsymbol{L}_\gamma(f)) \tag{10}$$

with $\boldsymbol{L}_\gamma(f)$ the weighted Laplacian of the hallucinated graph to a given frontier.

Fig. 4. View in Gazebo of the AWS bookstore (left) and house (right) scenarios.

The above optimization (which entails the graph hallucination and Hessian estimation processes) implicitly penalizes visiting distant candidates and encourages a balance between the uncertainty decrease in both the robot location and the map that occurs when reobserving known landmarks, and the increase of knowledge about the environment when visiting new regions.

Since the global plan was already computed, navigation comes down to following that path, a task inherent of the local planner. We use the time elastic band approach [19], which optimizes the trajectory locally with respect to different constraints. As occurred with the global planner, the local planner acts over a cost map built from lidar measurements.

3.5   Recovery Behavior

As shown in Fig. 1, navigation goals can be exceptionally obtained from the recovery module if ORB-SLAM gets lost during exploration. In such cases, the robot uses the wheel odometry to localize itself and generates navigation goals to previously-visited areas in order to facilitate relocalization. Since most usual tracking losses are due to getting excessively close to an obstacle, the first goal consists of a 180° rotation. If no relocalization occurs, the robot is commanded to previously-visited locations with high relocalization potential. To identify them, we search all the pose-graph nodes within a 2 m radius and compute the number of map points visible from each of them. The preferred destination will be that with highest map point density.

4   Experimental Validation

The system has been tested in Gazebo simulator. The experiment consists of autonomously exploring AWS bookstore and house scenarios[2], which texture is rich enough to be processed by visual SLAM algorithms; see Fig. 4. Termination condition is the absence of candidate frontiers. We used a wheeled robot with a Kinect RGB-D camera (for SLAM) and a lidar (for path planning and safe navigation). All experiments have been performed in Ubuntu 20.04 and ROS Noetic, using an Intel Core i9-10900K CPU and a NVidia GeForce RTX 3070
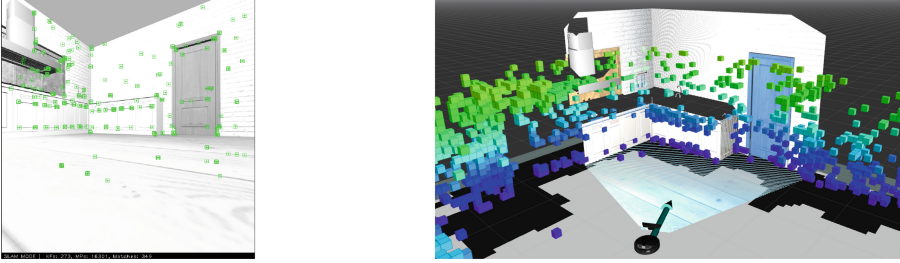
---

[2] https://github.com/aws-robotics/.

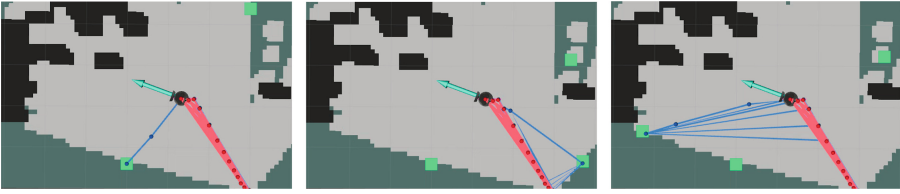**Fig. 5.** SLAM image with matched landmarks, Octomap and grid map built.

Fig. 6. Examples of the graph hallucination process during exploration.

GPU. For more details on the configuration parameters of each module, we refer the reader to the project repository.

To better show the proposed system, Figs. 5 and 6 contain two operating examples of the different modules during exploration of the house environment. In the former, we present the image fed to ORB-SLAM and the landmarks detected (left). Right image contains the Octomap built from those landmarks, on top of the depth image point cloud. For visualization purposes, Octomap height has been restricted between 0.1 and 2 m. Also, we include the rectified grid map and the estimated robot location (blue arrow). On the other hand, Fig. 6 illustrates the graph hallucination process towards three different frontiers (green squares). In all cases, the SLAM (red) and hallucinated (blue) pose-graphs have been plotted on top of the rectified grid map. While the first frontier does not produce reobservation edges in the hallucinated graph, the second and third do; showing that along the path to reach them, a greater number of known landmarks is expected to be observed.

Finally, Figs. 7 and 8 show the resulting maps and pose-graphs after the exploration of the two environments. In the first one, exploration lasted 560 s and ORB-SLAM was able to close two loops —thus performing two graph optimizations. In the second one, the time increased to 984 s and 5 optimizations were performed. Decision making represented only 5% and 7.3% of these times, respectively. Both environments were fully explored, despite the presence of open regions in the grid maps that correspond to low-textured walls in the simulator. Since planning feasibility is checked for all candidates, no frontiers were detected in these areas.
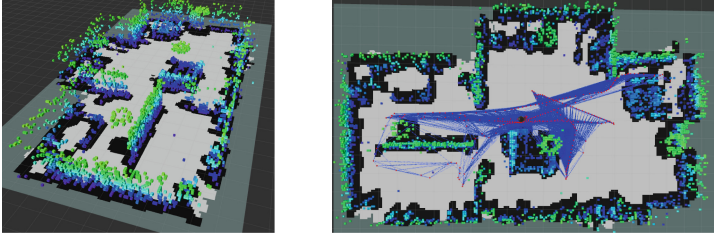
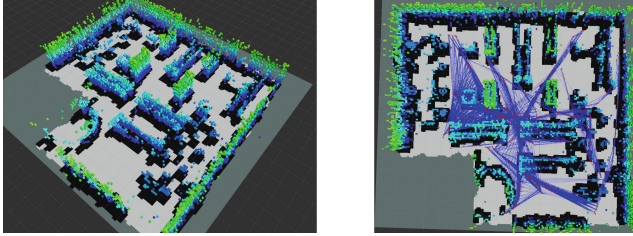**Fig. 7.** Maps and pose-graphs generated during exploration of house scenario.

Fig. 8. Maps and pose-graphs generated during exploration of bookstore scenario.

5   Conclusions

In this work, we have presented an open-source active SLAM method based on ORB-SLAM2, a visual graph-SLAM system that creates a landmark representation of the environment and accurately estimates the robot trajectory. We employ that information to perform decision-making over graphs, which involves: (i) identifying the candidate destinations, (ii) hallucinating the existing pose-graph towards those destinations and weighting their edges appropriately, and (iii) selecting the D-optimal candidate by analyzing the topology of the graphs. Experiments conducted on realistic simulated environments demonstrate the performance of the method. As future work, we plan to study multi-robot exploration behavior and conduct 3D experiments.

References

1. Placed, J.A., Strader, J., Carrillo, H., Atanasov, N., Indelman, V., Carlone, L., Castellanos, J.A.: A survey on active simultaneous localization and mapping: State of the art and new frontiers (2022). arXiv preprint arXiv: 2207.00254
2. Makarenko, A.A., Williams, S.B., Bourgault, F., Durrant-Whyte, H.F.: An experiment in integrated exploration. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 1, pp. 534–539 (2002)

3. Valencia, R., Miró, J.V., Dissanayake, G., Andrade-Cetto, J.: Active pose SLAM. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1885–1891 (2012)

4. Umari, H., Mukhopadhyay, S.: Autonomous robotic exploration based on multiple rapidly-exploring randomized trees. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1396–1402 (2017)

5. Carrillo, H., Dames, P., Kumar, V., Castellanos, J.A.: Autonomous robotic exploration using a utility function based on Rényi's general theory of entropy. Auton. Robot. **42**(2), 235–256 (2018)

6. Feder, H.J.S., Leonard, J.J., Smith, C.M.: Adaptive mobile robot navigation and mapping. Int. J. Robot. Res. **18**(7), 650–668 (1999)

7. Deng, D., Xu, Z., Zhao, W., Shimada, K.: Frontier-based automatic-differentiable information gain measure for robotic exploration of unknown 3D environments (2020). arXiv preprint arXiv:2011.05288

8. Carlone, L., Karaman, S.: Attention and anticipation in fast visual-inertial navigation. IEEE Trans. Rob. **35**(1), 1–20 (2018)

9. Wang, J., Englot, B.: Autonomous exploration with expectation-maximization. In: Amato, N., Hager, G., Thomas, S., Torres-Torriti, M. (eds.) Robotics Research, pp. 759–774. Springer, Cham (2020)

10. Placed, J.A., Castellanos, J.A.: Fast autonomous robotic exploration using the underlying graph structure. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 6672–6679 (2021)

11. Khosoussi, K., Huang, S., Dissanayake, G.: Novel insights into the impact of graph structure on SLAM. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2707–2714 (2014)

12. Khosoussi, K., Giamou, M., Sukhatme, G.S., Huang, S., Dissanayake, G., How, J.P.: Reliable graphs for SLAM. Int. J. Robot. Res. **38**(2–3), 260–298 (2019)

13. Placed, J.A., Castellanos, J.A.: A General Relationship between Optimality Criteria and Connectivity Indices for Active Graph-SLAM (2021). arXiv preprint arXiv:2110.01289

14. Chen, Y., Huang, S., Fitch, R.: Active SLAM for mobile robots with area coverage and obstacle avoidance. IEEE/ASME Trans. Mechatron. **25**(3), 1182–1192 (2020)

15. Mur-Artal, R., Tardós, J.D.: ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras. IEEE Trans. Rob. **33**(5), 1255–1262 (2017)

16. Kiefer, J.: General equivalence theory for optimum designs (approximate theory). Ann. Stat. **2**, 849–879 (1974)

17. Triggs, B., McLauchlan, P.F., Hartley, R.I., Fitzgibbon, A.W.: Bundle adjustment-a modern synthesis. In: Triggs, B., Zisserman, A., Szeliski, R. (eds.) International Workshop on Vision Algorithms, pp. 298–372. Springer, Heidelberg (1999)

18. Keidar, M., Kaminka, G.A.: Efficient frontier detection for robot exploration. Int. J. Robot. Res. **33**(2), 215–236 (2014)

19. Rösmann, C., Hoffmann, F., Bertram, T.: Integrated online trajectory planning and optimization in distinctive topologies. Robot. Auton. Syst. **88**, 142–153 (2017)