2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

Segmented DP-SLAM

Renan Maffei, Vitor Jorge, Mariana Kolberg, Edson Prestes
Instituto de Informática
Universidade Federal do Rio Grande do Sul
Porto Alegre - RS - Brazil
{rqmaffei, vamjorge, mariana.kolberg, prestes}@inf.ufrgs.br

***Abstract*— Simultaneous Localization and Mapping (SLAM) is one of the most difficult tasks in mobile robotics. While the construction of consistent and coherent local solutions is simple, the SLAM remains a critical problem as the distance travelled by the robot increases. To circumvent this limitation, many strategies divide the environment in small regions, and formulate the SLAM problem as a combination of multiple precise submaps. In this paper, we propose a new submap-based particle filter algorithm called Segmented DP-SLAM, that combines an optimized data structure to store the maps of the particles with a probabilistic map of segments, representing hypothesis of submaps topologies. We evaluate our method through experimental results obtained in simulated and real environments.**

I. INTRODUCTION

Simultaneous Localization and Mapping (SLAM) is the problem of building a map while dealing with uncertainty in localization. Solving the SLAM problem is a major requirement to the construction of real autonomous robots and has been the focus of current research [1], [2], [3].

Although many SLAM algorithms have been successfully presented, the SLAM remains particularly challenging when long distances have to be traversed by the robot. To circumvent this problem, many methods reduce the problem into solving low-level instances of SLAM. These methods, referred as submap-based SLAM, independently process limited regions of the environment, and later adjust the individual results to compose a solution.

A prominent approach of submap-based SLAM is to use hybrid maps, that is, combining a topological map with the metric map of the environment [4] [5] [6] [7]. In these strategies, the nodes of the graph that represent the topological map are associated to metric submaps, while its edges represent the connections between those submaps.

One of the earliest works with hybrid maps was the Atlas framework [4], where, as in most of the submap-based algorithms, the uncertainties of each submap are modeled according to its own coordinate system. Connections between submaps are detected by a matching process, and refined as the submaps uncertainties decrease.

Estrada et al. proposes an hierarchical approach [6]. Even though similar to Atlas, it introduces a loop closing technique that imposes consistency at the global level. Therefore, it increases the precision of the resulting global map.

Eliazar and Parr proposes another hierarchical approach [5], by extending the DP-SLAM algorithm [2] to a two-levels strategy. On the lower level, the DP-SLAM is performed inside a small portion of the map to obtain locally accurate paths. On the upper level, another instance of DP-SLAM is performed over the best resulting paths from the lower level. Since the upper level inputs are already refined, a smaller number of particles is required to obtain good results.

Blanco et al. presented the HMT-SLAM [7], which proposes a unified estimation of the hybrid metrical and topological path of the robot throughout the environment. Thus, the particles propagation is based not only on the motion and observation models of the robot, but also on a transition model associated to the topological map.

SegSLAM [8] introduces the idea of SLAM based on segments. A segment represents a limited region of the environment and is described by multiple submaps. Hence, different combinations of submaps (one for each segment), produce different solutions, i.e., different maps.

In this paper we propose a novel submap-based strategy, that combines the idea of environment segments containing multiples submaps, introduced by the SegSLAM algorithm, with an optimized data structure to store the maps of the particles. The distributed aspect of this structure, introduced by the DP-SLAM method [2], allows the segmentation of the environment in multiples submaps. The main contribution of this paper is the new submap-based SLAM algorithm for structured environments called Segmented DP-SLAM (SDP-SLAM). Other contributions are new approaches to estimate good combinations of map segments and to perform the matching of maps.

This paper is divided as follows. Section II presents the theoretical background, describing both the DP-SLAM and the SegSLAM. In Section III, we introduce the SDP-SLAM algorithm. In Section IV, we present some experimental results comparing our method to DP-SLAM and SegSLAM, in simulated and real environments. And finally, in Section V we draw the conclusions.

II. THEORETICAL BACKGROUND

A. DP-SLAM

DP-SLAM [2] is a Rao-Blackwellized particle filter (RBPF) algorithm that uses an optimized structure to store the maps of the particles. In RBPF-based strategies, the

978-1-4673-6357-0/13/$31.00 ©2013 IEEE

particles resampling process requires the copy of multiple instances of the map. Considering that, at each instant, the robot scans an area much smaller than the full map, the variation of each particle map between consecutive instants occurs only in a small region. DP-SLAM takes advantage of this fact to introduce an improved map representation. Basically, it merges all particles maps into only one map containing the differences observed by each particle, through a process called DP-Mapping.

DP-Mapping uses two efficient data structures: an ancestry tree and a modified grid map. The ancestry tree is the hereditary tree for all active particles of the filter. Its leaves represent the active particles, while the internal nodes are the ancestrals of these particles, i.e., the particles from which they derived. The second structure is a grid map that contains an observation tree for each cell. If a particle has an observation for a cell that is different from those made by its ancestrals, then the cell's tree is updated adding the information and observation of that particle. Thus to obtain the full map associated to a particle it is necessary to consult the observations made by the particle and by its ancestrals.

To ensure that the ancestry tree does not grow indefinitely, a pruning process is performed. When a particle does not generate a child, it is removed from the tree since its information will not be inherited. In addition, a particle that generates a single child has its information merged with the information of its child, to prevent the creation of branches without ramifications.

Despite all the space optimization, DP-SLAM still requires a very large number of particles to obtain good results.

B. SegSLAM

SegSLAM [8] is a submap-based SLAM approach that extends the particle filter estimation step to decide when a particle should stay in the current submap, re-enter an existent submap or move into a new submap. Differently from traditional RBPF SLAM, in SegSLAM, the particles are sampled from the distribution over, not only, poses, but also, submaps. The poses are described according the local coordinate system of the correspondent submap.

Another major distinction is that, while RBPF particles describe complete trajectory hypothesis, on SegSLAM, the particles are only responsible to generate submaps of the environment, which are stored in a structure called Segmented Map or SegMap. This structure maintains the connections between the submaps using a graph, and thus to reconstruct a possible robot trajectory it is necessary to concatenate compatible consecutive segments. This is done by sampling paths from the SegMap graph. The combinations of possible paths are weighted by a technique of submaps matching. Later, a list of potentially matchable segments is generated, with their respective positions transformations. In the end, particles choose matches from this list, or create new segments if a good matching has not been found.

Even though the increase in the diversity of solutions resulting from the numerous possibilities of segments combinations is beneficial to circumvent the problem of particle depletion, it considerably increases the search space. It is difficult to find a good combination of submaps comprising the entire robot trajectory when the number of segments grows. Hence, the sampling step in SegSLAM only considers a local analysis, i.e., the algorithm chooses the best samples from chains of few segments, not samples of the entire trajectory.

III. SEGMENTED DP-SLAM

Our proposed method called Segmented DP-SLAM, or simply SDP-SLAM, combines maps based on segments, from SegSLAM, to the particles ancestry tree, from DP-Slam. The idea is to capture the high diversity of solutions, ie. global maps generated by numerous possibilities of submaps combinations.

SDP-SLAM is based on the fact that observations made by the robot within small regions are highly related to one another, while distant observations are most likely not. Therefore, the segmentation of the environment in submaps is feasible. Following the formalizations of HMT-SLAM [7] and SegSLAM [8], we define the global map of the environment as

$$\Theta = \langle \{\theta_i\}_{i \in \Upsilon_t} \, , \, \{T_{a,b}\}_{a,b \in \Upsilon_t} \rangle \tag{1}$$

where Υ_t is the set of segments known at instant t. $\theta_i = \{\theta_i^{(1)}, \theta_i^{(2)}, \cdots, \theta_i^{(p)}\}$ is the set of all metric submaps associated to the i-th segment of Υ_t. Each submap belonging to θ_i is generated by one of the p particles of the bottom level filter. $T_{a,b} = \{T_{a,b}^{(1,1)}, T_{a,b}^{(1,2)}, \cdots, T_{a,b}^{(1,p)}, T_{a,b}^{(2,1)}, \cdots, T_{a,b}^{(p,p)}\}$ are the coordinate transformations of all the possible combinations between submaps of two adjacent segments a and b.

The state $\mathbf{s}_t$ of the robot pose at instant t is given by

$$\mathbf{s}_t = \langle \mathbf{x}_t \, , \, \gamma_t \rangle \tag{2}$$

where $\mathbf{x}_t$ represents the metric robot pose, while γ_t indicates to which submap the robot pose is associated.

Knowing these definitions, the posterior distribution of SLAM considering $\mathbf{s}_{1:t}$ and Θ is defined as

$$p(\mathbf{s}_{1:t}, \Theta \,|\, \mathbf{z}_{1:t}, \mathbf{u}_{1:t}) \tag{3}$$

where $\mathbf{z}_{1:t}$ and $\mathbf{u}_{1:t}$ are, respectively, the sets of observations and actions made by the robot.

The SDP-SLAM structure overview is presented on Figure 1. SDP-SLAM is composed of two levels of particle filters. The bottom level process is responsible to estimate the segments of the robot path $\mathbf{x}_{1:t}$, the associated local maps $\theta_{1:t}$ and the transformations between submaps $T_{1,2:t-1,t}$. The top level process is responsible to estimate the topology of the global map, in other words, find the best combinations of submaps $\gamma_{1:t}$.

At the bottom level, the operation of the particle filter is similar to the filter in SegSLAM, where the particles are only responsible to generate locally accurate submaps. The key difference lies in the way the maps and trajectories are constructed, using the DP-Mapping process.

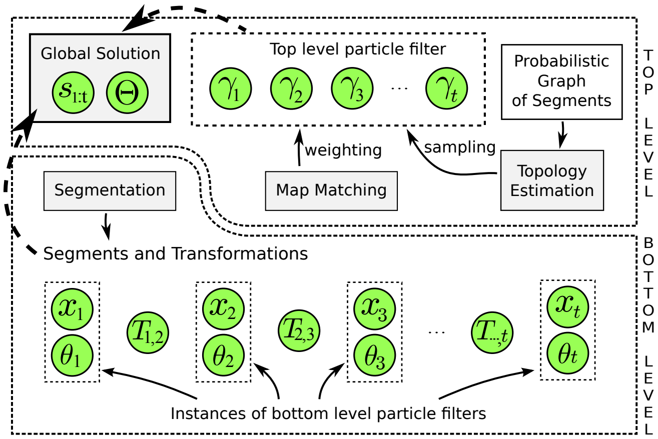


Fig. 1. SDP-SLAM structure overview.

At the top level, a particle filter is used to estimate good combinations of submaps of the environment. The propagation of the particles is made by the particles transition model, which is a function of the prior set of particles and the set of all actions and observations made by the robot:

$$\gamma_{1:t} = q(\gamma_{1:t} \,|\, \gamma_{1:t-1}, \mathbf{z}_{1:t}, \mathbf{u}_{1:t})$$

This transition model uses a Probabilistic Graph of Segments (PGS), that contains weighted connections between submaps of adjacent segments. PGS represents a Monte Carlo approximation of the probability distribution of all possible topologies that can represent the environment. As said, a segment of the environment contains multiple submaps, each one built by a different particle. Those submaps are represented by nodes of the PGS, which are grouped in levels representing the segments. The connections between submaps of adjacent segments are the edges of the PGS.

The SDP-SLAM algorithm is presented in Algorithm 1. The first step (line 1) is the initialization of both particle filters. At the bottom level, all particles start at the same position with empty submaps. At the top level, all particles start with empty submaps combinations, since no segmentation was performed yet.

In the main loop, the first step (line 2) is the acquisition of the odometry and sensors measurements. Next, the update of the bottom level filter is performed to build hypotheses of local maps. The initial steps are the same performed by DP-SLAM (line 3). First, particles are propagated inside the current segment and weighted. Then, the resampling is made, the ancestry tree is updated through a process of prune and merge, and the particles observations are updated into the Distributed Particle Map.

Next, occurs the segmentation decision (line 4). Most of submap-based SLAM approaches performs the segmentation practically on a regular time interval or according to error measures [4] [9]. Our method was tested with a periodic segmentation at a fixed time and a segmentation based on the particles dispersion.

Whenever a segmentation occurs, the bottom level particle filter is stopped and the ancestry tree section regarding the

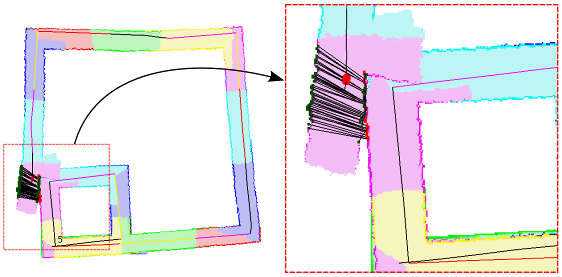
Algorithm 1: SDP-SLAM algorithm

1 **Initialization**
while *the robot is navigating* **do**
2   **Read odometry and sensors measurements**.
  Bottom level process:
  begin
3   **DP-SLAM update**
4   **Segmentation decision**
    if *a segmentation occurs* **then**
5   **Ancestry tree anchorage**
6   **Particles restart**
    end
  end
  Top level process:
  begin
    if *a segmentation occurs* **then**
7   **Insertion of a new level of nodes in the PGS**
    end
8   **Estimation of submaps combinations**
9   **Weighting of submaps combinations**
10   **Update of the PGS**
  end
end

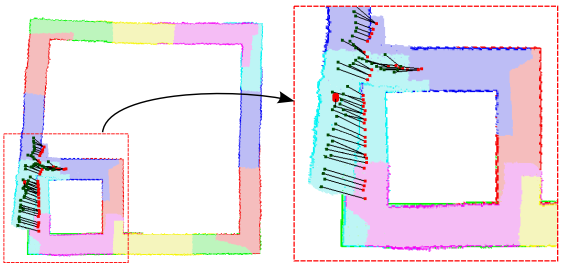
last segment is anchored, so it cannot be modified later (line 5). The current set of particles is restarted to allow the construction of new independent submaps (line 6). At the top level, a new set of nodes representing the submaps of the new segment is inserted into the PGS. Among the information stored in those nodes are the identification of the particle, required for queries on ancestry tree; and the initial and final transformations of the submap (the first and the last robot poses of the submap), used to combine submaps in a same coordinate system.

The next step is the update of the top level particles (line 7), that are responsible for estimating hypotheses of submaps combinations. We adopt an elitism strategy, so a set of the best particles is maintained, while a set of the worst is eliminated in the resampling step. The remaining particles are the ones responsible by the diversification of the top level filter. The submaps combinations associated to these particles are modified according to queries on the PGS. (How the probabilities in the PGS are defined will be explained in the last step of the algorithm.)

Then, the evaluation of the top level particles is performed through the matching between overlapping submaps of each sample (line 8). As in SegSLAM, the matching process is made with the ICP (Iterative Closest Point) algorithm [10]. ICP is a very simple and fast method, but requires that the two sets of points being compared have a strong association, otherwise, the method might converge to local minimum or even not converge. In general, ICP uses the information about obstacles, disregarding the information about empty spaces implied by the use of range sensors. However, when the alignment error of segments is too large, the association of points might be incorrect, as shown in Figure 2(a), where points of a wall were associated to points of a wrong wall. In our case, as shown in (b), we select points from the middle of



(a) Extracting points from obstacles



(b) Extracting points from the middle of free-space

Fig. 2. Comparison between the matching of submaps extracting points from obstacles and from the middle of free-space.

free-space regions, like the map merging strategy of Saeedi et al. [11]. Using these points we reduce ambiguities like the illustrated in (a).

The last step of our method is the update of the PGS (line 9). Systematically, samples of submaps combinations are generated and evaluated by the matching process using ICP. The result of the matching is a measure of the ICP error. The measured error of each sample is added to the accumulated errors of the connections between submaps that compose that sample. For example, the accumulated error E_{1b2a} of the connection between γ_{1b} and γ_{2a} is the sum of the errors from all sampled combinations having γ_{1b} and γ_{2a}. The idea is that, over time, connections with low accumulated errors possess great chance to compose good solutions. So, the probability to choose a pair of submaps is inversely proportional to the accumulated error of this association. We compute the inverse of the accumulated error and normalize the values to obtain probabilities.

$$p(\gamma_{1b}, \gamma_{2a}) = \frac{\sum_{i,j=1}^{p} 1/E_{1i2j}}{1/E_{1b2a}} \tag{4}$$

Figure 3 shows an example of the functioning of SDP-SLAM. In (a), three submaps of a same segment are depicted in the map. At this point, the ancestry tree contains only the particles of the first segment, as shown in (b), while the PGS only have the three nodes (submaps 1A, 1B, 1C) of the first segment, as shown in (c). When the method is processing the second segment, in (d), the ancestry tree contains the particles of the first and the second segments, as shown in (e). In (f), the graph of segments contains two levels of nodes, representing both segments. Then, as shown in (g), it is possible to combine the submaps of the two segments and evaluate the sampled combinations. In (h), we open a parenthesis to show that the ancestry tree continues to be pruned, like in DP-SLAM. In (i), the weight of each connection between submaps is updated in the graph of segments. These estimated weights will be used during the sampling step of the top-level particle filter. Finally, in (j), a possible trajectory is reconstructed by combining two adjacent submaps ($1A$ and $2C$). The global map is built by consulting the observations made by each particle of the selected branches of the ancestry tree, highlighted in (k). A transformation T must be applied to put both submaps in the same coordinate system, as shown in (l). This transformation is a composition of the final pose from the first submap with the initial pose from the second submap.

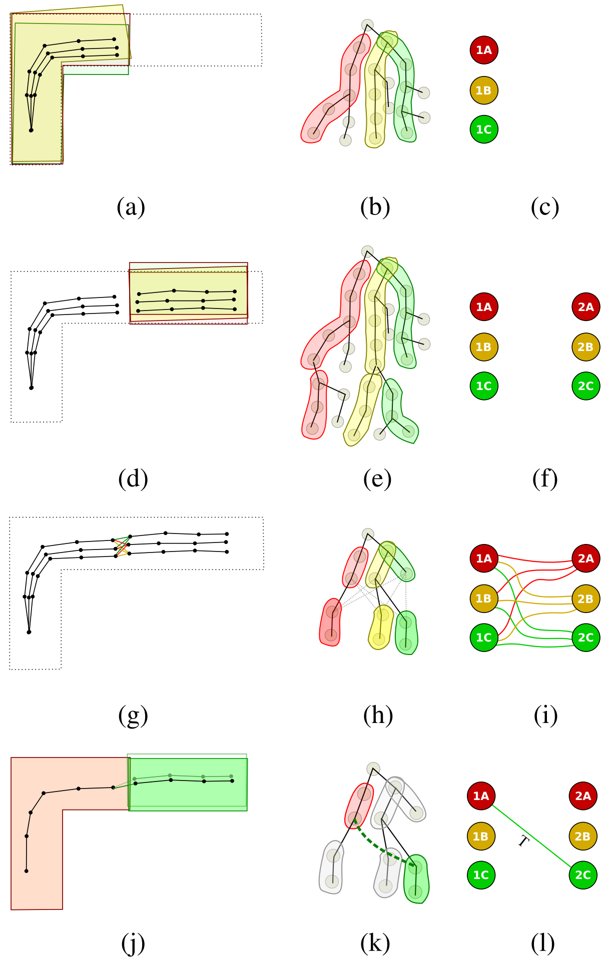


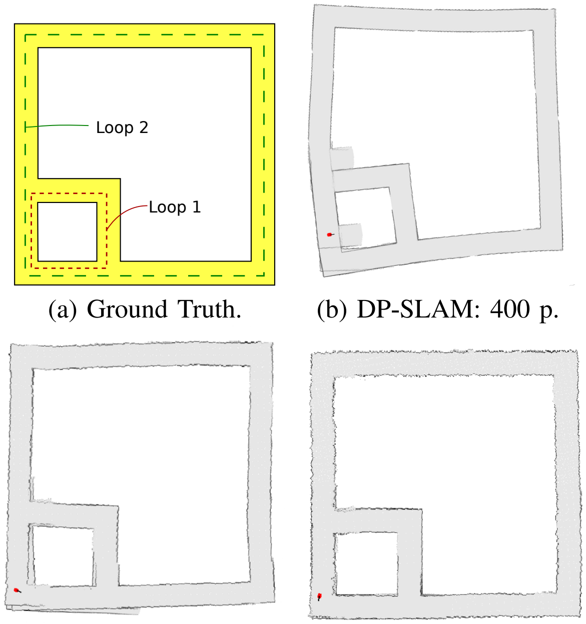
Fig. 3. Example of functioning of SDP-SLAM.

IV. Experiments

The evaluation of SDP-SLAM was made through experiments in simulated and real environments, that are illustrated in Figure 4(a) and Figure 5(a) respectively. The simulated environment contains an inner loop (loop 1) and an outer loop (loop 2) with lenghts of $28m$ and $80m$, respectively. The real environment contains three loops, corresponding to corridors of a building from the Institute of Informatics at UFRGS. The two inner loops have lenghts of $43m$ (loop 1) and $57m$ (loop 2), and together form a larger loop of $88m$ (loop 3).

We choose these environments because they contain nested loops that aggravate the particle depletion problem. For instance, during the mapping of an inner loop, a RBPF strategy discard particles that do not have the highest weights, but that can be needed later to map an outer loop.

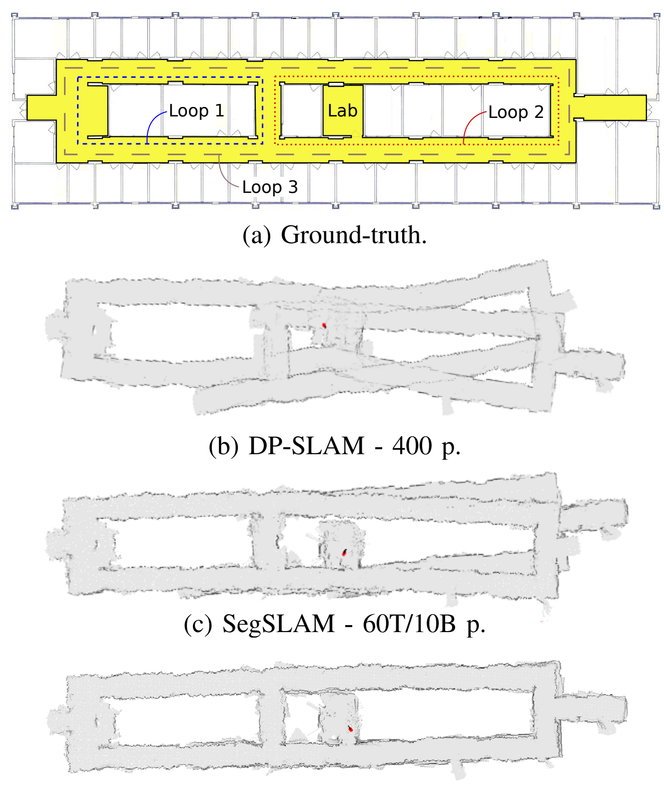
We compared our approach to SegSLAM and DP-SLAM, the methods which served as the basis for SDP-SLAM. Regarding the comparison with SegSLAM, we focussed on the topology estimation step to highlight the differences in the construction of the global map, keeping the remainder of the process (such as the matching and the segmentation) the same of SDP-SLAM. On the other hand, DP-SLAM and SDP-SLAM could not be compared directly. SDP-SLAM is a hierarchical method with two different particle filters, while DP-SLAM has only one. In order to perform a fair comparison between the methods, we decided to use as many particles as possible considering the same running times.



(a) Ground Truth. (b) DP-SLAM: 400 p.

(c) SegSLAM: 60T/10B p. (d) SDP-SLAM: 60T/10B p.

Fig. 4. Comparison between SDP-SLAM, DP-SLAM and SegSLAM through experiments in the simulated environment.



(a) Ground-truth.

(b) DP-SLAM - 400 p.

(c) SegSLAM - 60T/10B p.

(d) SDP-SLAM - 60T/10B p.

Fig. 5. Comparison between SDP-SLAM, DP-SLAM and SegSLAM through experiments in the real environment.

Figure 4 shows the resulting maps of the experiments in simulated environment, where the robot is the red point, the obstacles are black and the free-space is gray. In (b), we present the map built with DP-SLAM using 400 particles. With this number of particles, the method did not properly close the larger loop. In (c), we show the map built using the SegSLAM topology estimation step with 60T/10B particles (60 samples of submaps combinations at the top level and 10 particles to construct the submaps at the bottom level). The resulting map is good, but it presents some inconsistencies. Finally, in (d), the result of SDP-SLAM with 60 particles at the top level and 10 at the bottom level is shown. As it can be observed, the resulting map is the closest to the environment ground truth.

The results in real environment are shown in Figure 5. In this environment the path traversed by the robot was larger than in the simulated one, thus the methods ended up having more difficulties, as well as the difference between the results became more visible. The map built by DP-SLAM using 400 particles is shown in (b), and the result is very poor, since DP-SLAM was not able to close any loop. As shown in (c), using the SegSLAM topology estimation, with the same configuration of 60T/10B particles, the result was better but not satisfying. Comparatively, the map presented in (d), obtained by SDP-SLAM using 60T/10B particles, is visually better than the maps produced by the other algorithms.

Besides the visual comparisons, we also made the evaluation of the topology estimations by measuring the mean alignment error of the solutions, in meters, during the SLAM process. This measure is given by the mean ICP nearest neighbor error computed in the submaps matching process. Figures 6 and 7 show the mean error variation from the solutions of the experiments in simulated and real environment, respectively. All lines in these figures have straight line segments, because the submaps matching process is only performed when there are overlapping submaps, that is, during periods in which the robot is closing loops. In the rest of the process the weights of the samples are not changed.

Figure 6 shows that, at the closure of the small loop (loop 1), the mean error is low ($< 0.5m$) for any of the settings

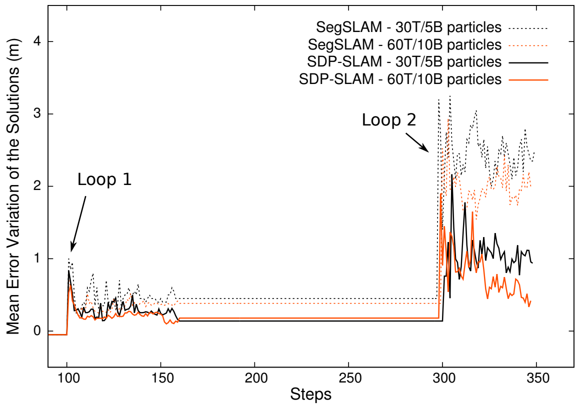


Fig. 6. Mean ICP error variation from the solutions of the experiments in simulated environment.

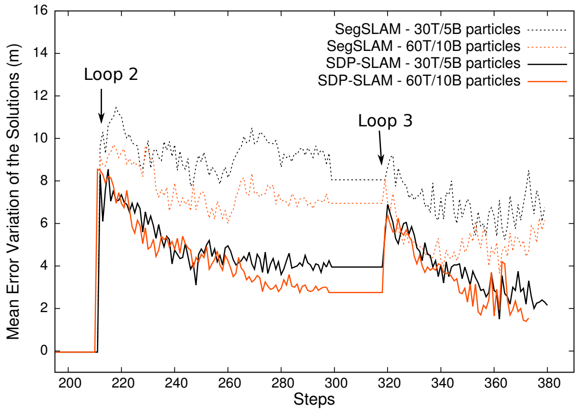


Fig. 7. Mean ICP error variation from the solutions of the experiments in real environment.

used. The big problem happens when the robot returns to its initial position, after covering the major loop (loop 2). At first, the error is large, because the overlap among submaps is not good in the beginning of loop closures. Later, using the strategy adopted by SegSLAM, the error oscillates around $2m$, while with SDP-SLAM the error continues to decrease below $1m$.

In Figure 7, we observe that the errors reach values nearly three times higher than in the simulated environment. This can be explained by a couple of factors. First, the robot motion model is more inaccurate in the real environment, and second, the environment is larger, and therefore more difficult to map. Also, the robot started its path traveling around the larger inner loop (loop 2), instead of the minor loop (loop 1). This situation led to a mean error bigger in the beginning of the experiment than in the rest.

The mean and standard deviation of the ICP error during the experiments in simulated and real environments are shown in Table I. The error is smaller using the SDP-SLAM topology estimation than using the SegSLAM topology estimation. The best results were obtained with the configuration of 60T/10B particles in SDP-SLAM.

	SegSLAM				SDP-SLAM			
	Simulated		Real		Simulated		Real	
Particles	μ	σ	μ	σ	μ	σ	μ	σ
30T/5B	1.43	0.84	8.41	2.20	0.66	0.34	4.50	1.37
30T/10B	1.26	1.04	7.55	2.24	0.58	0.36	4.28	1.48
60T/5B	1.23	1.18	7.04	2.48	0.61	0.40	4.12	1.53
60T/10B	1.15	1.22	6.63	2.56	0.48	0.45	3.86	1.62

TABLE I

MEAN AND STANDARD DEVIATION OF THE ICP ERROR IN SEGSLAM AND SDP-SLAM DURING THE EXPERIMENTS

V. CONCLUSION

The results obtained in the experiments showed that SDP-SLAM generates better solutions than the original DP-SLAM, using a much smaller number of particles. We also performed experiments comparing SDP-SLAM to SegSLAM. The evaluation of the topology estimation process showed that our method indeed searches for solutions with low alignment errors. As measured in the experiments, the error associated to the samples of submaps combinations tends to decrease over time.

As a future work, we intend to improve SDP-SLAM by making the update of submaps considering the information associated with other submaps. This will probably improve the quality of the submaps, but will reduce the possibilities of submaps combinations. An idea is to apply this strategy only when a set of submaps are well established (eg. after closing a perfect loop). Thus, such submaps would be permanent components of the global solutions.

REFERENCES

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

[2] A. Eliazar and R. Parr, "Dp-slam: fast, robust simultaneous localization and mapping without predetermined landmarks," in *Proceedings of the 18th international joint conference on Artificial intelligence*, ser. IJCAI'03, 2003, pp. 1135–1142.

[3] G. Grisetti, R. Kummerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based slam," *Intelligent Transportation Systems Magazine, IEEE*, vol. 2, no. 4, pp. 31–43, winter 2010.

[4] M. Bosse, P. Newman, J. Leonard, M. Soika, W. Feiten, and S. Teller, "An atlas framework for scalable mapping," in *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, vol. 2, sept. 2003, pp. 1899–1906.

[5] A. I. Eliazar and R. Parr, "Hierarchical linear/constant time slam using particle filters for dense maps," in *Advances in Neural Information Processing Systems 18*, 2006, pp. 339–346.

[6] C. Estrada, J. Neira, and J. D. Tardos, "Hierarchical slam: Real-time accurate mapping of large environments," *Robotics, IEEE Transactions on*, vol. 21, no. 4, pp. 588–596, aug. 2005.

[7] J. L. Blanco, J. A. Fernandez-Madrigal, and J. Gonzalez, "Toward a unified bayesian approach to hybrid metric–topological slam," *Robotics, IEEE Transactions on*, vol. 24, no. 2, pp. 259–270, april 2008.

[8] N. Fairfield, G. A. Kantor, and D. Wettergreen, "Segmented slam in three-dimensional environments," *Journal of Field Robotics*, vol. 27, no. 1, pp. 85–103, February 2010.

[9] T.-k. Lee, S. Lee, and S.-y. Oh, "A hierarchical rbpf slam for mobile robot coverage in indoor environments," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, sept. 2011, pp. 841–846.

[10] N. Fairfield and D. Wettergreen, "Evidence grid-based methods for 3d map matching," in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, may 2009, pp. 1637–1642.

[11] S. Saeedi, L. Paull, M. Trentini, M. Seto, and H. Li, "Efficient map merging using a probabilistic generalized voronoi diagram," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, oct. 2012, pp. 4419–4424.