Merging Partially Consistent Maps

Taigo Maria Bonanni, Giorgio Grisetti, and Luca Iocchi

Dept. of Computer, Control and Management Engineering
Sapienza University of Rome
Via Ariosto 25, I-00185, Rome, Italy
{bonanni,grisetti,iocchi}@dis.uniroma1.it

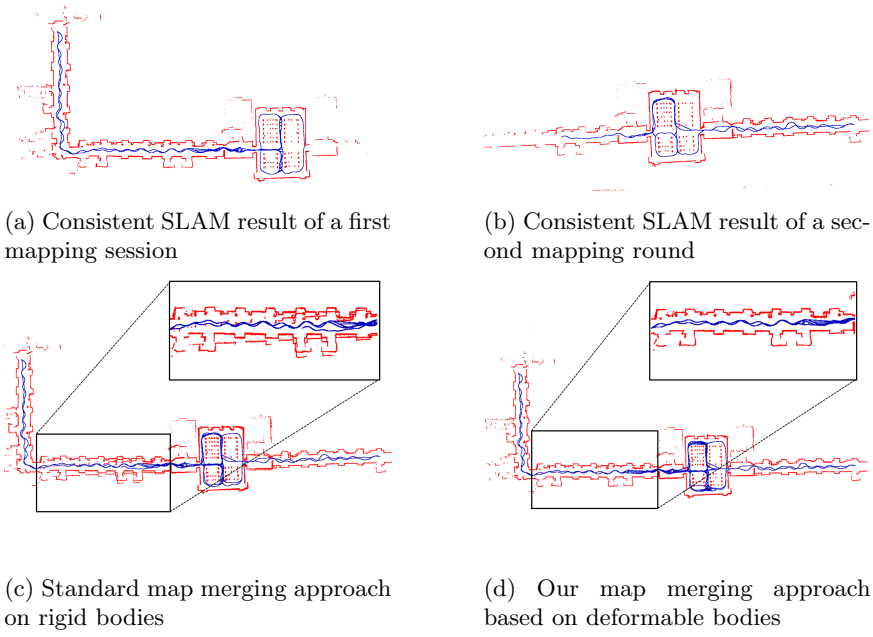
Abstract. Learning maps from sensor data has been addressed since more than two decades by Simultaneous Localization and Mapping (SLAM) systems. Modern state-of-the-art SLAM approaches exhibit excellent performances and are able to cope with environments having the scale of a city. Usually these methods are entailed for on-line operation, requiring the data to be acquired in a single run, which is not always easy to obtain. To gather a single consistent map of a large environment we therefore integrate data acquired in multiple runs. A possible solution to this problem consists in merging different submaps. The literature proposes several approaches for map merging, however very few of them are able to operate with local maps affected by inconsistencies. These methods seek to find the global arrangement of a set of rigid bodies, that maximizes some overlapping criterion. In this paper, we present an off-line technique for merging maps affected by residual errors into a single consistent global map. Our method can be applied in combination with existing map merging approaches, since it requires an initial guess to operate. However, once this initial guess is provided, our method is able to substantially lessen the residual error in the final map. We validated our approach on both real world and simulated datasets to refine solutions of traditional map merging approaches.

1 Introduction

To autonomously execute complex tasks such as object delivery, house cleaning, etc., mobile robots need to know their operating environment. This is usually addressed by the Simultaneous Localization and Mapping (SLAM) problem, that provides an estimate of the map and of the robot trajectory based on the robot measurements.

SLAM has been object of research for more than two decades, and effective solutions are available [21],[18],[9],[22],[16],[19]. The most common sensor used to build robotic maps is the laser scanner, and existing approaches can effectively be used to construct maps.

Regardless the technique employed, unless one uses some sort of absolute sensor like a GPS, even the most effective methods might fail when the environment size becomes too big. Acquiring data for SLAM is an error-prone procedure that requires attention to obtain satisfying results. Often the operator forgets to



(a) Consistent SLAM result of a first mapping session

(b) Consistent SLAM result of a second mapping round

(c) Standard map merging approach on rigid bodies

(d) Our map merging approach based on deformable bodies

Fig. 1: Motivating example of our approach. *a), b)* are the input graphs. The global map shown in *c)* is affected by inconsistencies that are corrected by our algorithm, as shown in *d)*.

visit certain locations during data acquisition, or makes mistakes that challenge the SLAM engine he is using (e.g. in a VisualSLAM session one might enter in a room with poor light conditions).

Furthermore, typical SLAM systems require to operate on data gathered in a single session. In other words they require a continuous trajectory to produce a consistent map. If the environment changes, a new map needs to be computed from new data. This results in the need of performing tedious and error prone data acquisition sessions, each time the environment changes. A branch of literature addresses this problem by employing a team of robots, instead of a single one, thus addressing the multi-robot SLAM problem. Alternatively one might merge different local maps, acquired at different points in time. If the sole aim is to get the map, the latter approach has obvious advantages, since it requires less resources. Furthermore, in case the environment changes, new data have to be acquired only in the changed portion, and be subsequently integrated in the global map, after removing the outdated local map.

In this paper, we present a novel approach for merging partially consistent maps. Typical map merging methods regard the maps as rigid bodies. A solution for map merging is a set of rigid transformations between the maps that maximizes their consistent overlap. This has the obvious limitation of not dealing with noise that might affect the local maps, and result in artifacts. This error is always present and comes from the process used to estimate the input

maps. Conversely, our method operates on maps that are regarded as network of springs and masses. We reduce the problem of merging two maps to the problem of deforming two networks onto each other so that the residual energy of the system is minimal. This corresponds to finding the configuration of the nodes in the two networks that best preserves the original layout of the input maps. Notice that the algorithm presented in this work is orthogonal to existing map merging methods, and can be used to refine their solutions when the input maps are affected by substantial error.

In case one uses the well known graph-based formulation of SLAM [8], such a network is already provided as a pose graph. If the map is only available as an occupancy grid or an image, we detail how to obtain a feasible pose graph using Voronoi diagrams. By operating on graphs, our method benefits from a reduced dimensionality, thus achieving efficient computation while preserving the details of the input data. In Figure 1 we provide a motivating example for our work. Fig, 1*a)* and 1*b)* are the SLAM results of two different mapping session, performed in the same environment. We want to merge these maps into a single one, possibly minimizing the error. Since each map is affected by residual errors, combining them through a single rigid transformation propagates the error in the global map, as shown in Fig. 1*c)*. Instead, our approach, illustrated in Fig. 1*d)*, can achieve the goal by deforming the maps onto each other. This has the dual effect of increasing the global consistency and of providing a consistent map for navigation.

2 Related Work

State-of-the-art solutions to map merging are based on image registration techniques. The partial maps are regarded as tiles of an image that should be composed to form the global map. Erinc *et al.*, [7], present an anytime merging technique based on appearance maps, exploiting image similarity to find candidates. Other approaches, [20], [5], are based on the spectral analysis of the Hough transforms of the maps to merge. These methods aim at finding the position of each local map with respect to the frame of the global map. A shortcoming of this class of algorithms is their sensitivity to errors in the local maps. When these local maps are subject to deformations, a set of pure rigid transformations is not sufficient to compute a globally consistent map. Our method is orthogonal to these approaches. In our current implementation we did not focus on providing an initial alignment between the maps. This task can be solved by using one of the methods above. Conversely, our approach seeks to maximize the consistency of partially aligned maps, by warping them onto each other.

Map merging has been addressed also in the context of multi-robot SLAM. Léon *et al.* proposed an approach based on particle filters. Their SLAM system merges the maps of different robots by using the map estimated by one robot as a measurement in the filter of another robot. Carlone *et al.*, [4], present an approach using an estimation kernel to compute the relative transformation among robots, without any a-priori knowledge. Leung *et al.*, in [15], presented a decen-

tralized and distributed algorithm for cooperative SLAM, based on a set of rules for the interchange of information. Cunningham *et al.*, in [6], presented an approach called Decentralized Data Fusion (DDF) for the generation of a common map called *condensed map* built after the marginalization of shared landmarks. Lazaro *et al.*, [14], presented a multi-robot SLAM approach based on condensed measurements for the exchange of information among robots. These methods are designed to operate within a full SLAM pipeline which requires to cope with on-line aspects and to deal with communication issues. In the context of multi-agent systems, Jennings *et al.* [12] presented a distributed mapping approach that relies on an approximation of generalized Voronoi graphs, built during the mapping procedure, and a simple distance-based metric for matching portions of graph. Topological approaches address the problem framing it as a graph isomorphism, where one wants to find the set of equivalences between nodes and edges of different subgraphs [3]. Along this line of research, Huang *et al.* [11], presented an approach merging topological maps inspired to maximal common subgraph problem. After building an initial set of matching hypotheses among vertices of the graphs, these hypotheses are then discarded or confirmed considering geometric features of the environment. These methods, however require a symbolic representation of the map, which is not provided by common SLAM engines.

In summary, existing map merging methods aim at getting a global alignment between local maps, but they are highly sensitive to noise in the input. Multi robot SLAM systems are able to deal with this problem, but they are rather complex and, in order to operate correctly, they require the data to be acquired at the same time. In this paper we present an approach to generate a globally consistent map out of noisy local maps.

3 Graph-Based Map Merging

Our approach is an off-line procedure for merging partially consistent local maps having a limited overlap. The most suitable representation for our purpose is to model the maps using graphs of measurements, as in the graph-based SLAM formalization [8]. To align the maps into a globally consistent one, we iteratively deform one of them onto the other, connecting robot poses, which belong to different graphs, if the observations made at these nodes are similar.

The remainder of this section is organized as follows. In Section 3.1, we detail our formalization of maps as deformable bodies. Since graphs of measurements from SLAM are not always available, we propose a method for the extraction of deformable networks out of occupancy grid maps in Section 3.2. Subsequently, in Section 3.3, we address how the inter-graph data association is carried out.

3.1 Map Representation

According to the graph-based SLAM formalization, we represent a deformable map as a graph of the form: $\mathcal{G} = \langle \mathcal{X}, \mathcal{C} \rangle$. Each node $\mathbf{x} \in \mathcal{X}$ is associated to

a robot pose in the environment. Each edge $e = \langle \mathbf{x}_i, \mathbf{x}_j \rangle \in \mathcal{C}$ expresses spatial relationships among poses, either from the robot motion or by correlating similar observations of the environment. Since the nodes of this graph are robot poses, they are also called *pose graphs*.

The energy or log likelihood of a configuration of nodes can be formulated as follows. Let $\mathbf{x} = (\mathbf{x}_1, \ \dots \ , \mathbf{x}_n)^T$ be a vector of poses, where $\mathbf{x}_i$ describes the pose of node i. Let $\mathbf{z}_{ij}$ and $\mathbf{\Omega}_{ij}$ be respectively the mean and the information matrix of an edge between the node i and the node j. In energetic terms, $\mathbf{z}_{ij}$ can be seen as equilibrium point of a spring with stiffness $\mathbf{\Omega}_{ij}$ connecting the masses located at $\mathbf{x}_i$ and $\mathbf{x}_j$. If this edge represents a virtual measurement, its equilibrium point is the transformation that makes the observation acquired from i maximally overlap with the observation acquired from j. Let $\hat{\mathbf{z}}_{ij}(\mathbf{x}_i, \mathbf{x}_j)$ be the relative transformation between the two nodes. For a given configuration the energy l_{ij} of a measurement $\mathbf{z}_{ij}$ is therefore

$$\mathrm{l}_{ij} \propto [\mathbf{z}_{ij} - \hat{\mathbf{z}}_{ij}(\mathbf{x}_i, \mathbf{x}_j)]^T \mathbf{\Omega}_{ij} [\mathbf{z}_{ij} - \hat{\mathbf{z}}_{ij}(\mathbf{x}_i, \mathbf{x}_j)]. \tag{1}$$

Let $\mathbf{e}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{z}_{ij})$ be a function that computes the distance from the two poses to the equilibrium point. For simplicity of notation, we will encode the indices of the measurement in the indices of the error function

$$\mathbf{e}_{ij}(\mathbf{x}_i, \mathbf{x}_j) \ = \ \mathbf{z}_{ij} - \hat{\mathbf{z}}_{ij}(\mathbf{x}_i, \mathbf{x}_j). \tag{2}$$

Minimizing the energy of the graph consists in finding the configuration of the nodes $\mathbf{x}^*$ that minimizes the energy of all the edges

$$\mathbf{F}(\mathbf{x}) = \sum_{\langle i,j \rangle \in \mathcal{C}} \mathbf{e}_{ij}^T \mathbf{\Omega}_{ij} \mathbf{e}_{ij}, \tag{3}$$

thus, it seeks to solve the following equation:

$$\mathbf{x}^* = \operatorname*{argmin}_{\mathbf{x}} \mathbf{F}(\mathbf{x}). \tag{4}$$

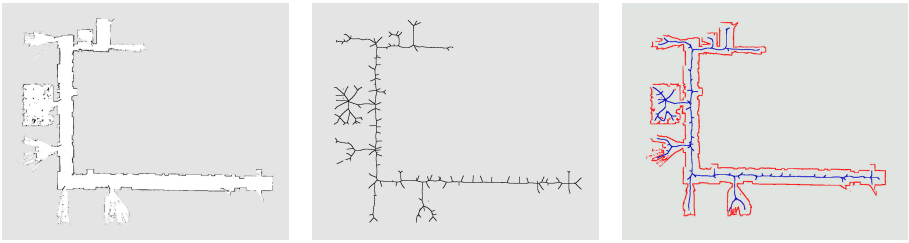
A graph-based SLAM engine will already provide a graph in a minimal energy configuration. However, when merging two graphs, the individual solutions might not be globally optimal due to the addition of constraints between different graphs. Thus, to find the most likely configuration, we need to compute a new assignment of poses that minimizes the following equation:

$$\mathbf{F}(\mathcal{X}) = \sum_{\langle i,j \rangle \in \mathcal{C}_1} \mathbf{e}_{ij}^T \mathbf{\Omega}_{ij} \mathbf{e}_{ij} + \sum_{\langle i,j \rangle \in \mathcal{C}_2} \mathbf{e}_{ij}^T \mathbf{\Omega}_{ij} \mathbf{e}_{ij} + \sum_{\langle i,j \rangle \in \mathcal{C}_{12}} \mathbf{e}_{ij}^T \mathbf{\Omega}_{ij} \mathbf{e}_{ij} \tag{5}$$

Here $\mathcal{C}_1$ and $\mathcal{C}_2$ are respectively the edges in the first and the second graphs, while $\mathcal{C}_{12}$ are the edges connecting the first and the second graph. To solve this problem we use the open source **g2o** optimization package [13]. The effects of potential outliers is reduced by using Dynamic Covariance Scaling [1] to lessen the contribution of edges that disagree with their neighbors.

3.2 Obtaining the Representation from Grid Maps

If we have only sets of grid maps, we can still generate the required pose graphs, complete of measurements, by using Voronoi diagrams. The Voronoi diagram



(a) Input grid map (b) Extracted Voronoi diagram (c) Output graph of measurements

Fig. 2: Extracting the Voronoi Diagram out of an input grid map, we are able to generate a graph of measurements compliant with our graph matcher.

can be straightforwardly extracted as the locus of points in the free space that are equidistant from at least two occupied cells in the map. On grid maps, the Voronoi diagram provides also a good estimate of the topology of the environment and a plausible trajectory for the robot. The pose graph is computed sampling points of the diagram and correlating nearby nodes according to its connectivity. Finally, the observations are obtained by ray-tracing the obstacles of the grid map to the sampled poses, simulating the behavior of a laser scanner. The typical output of the procedure is shown in Figure 2.

3.3 Data Association among Partially Consistent Maps

In this section we describe our graph-merging procedure. The input of our algorithm are two graphs $\mathcal{G}_r$ and $\mathcal{G}_m$, respectively the *reference graph* and the *matchable graph*. The output is a set of edges $\mathcal{M}$ connecting vertices of $\mathcal{G}_r$ to vertices of $\mathcal{G}_m$. We assume that the two graphs have been partially overlayed in a region around an initial node $\mathbf{x}_r$ of the matchable graph. We progressively deform the matchable graph, towards the reference, according to a breadth first visit. Each time we expand a node $\mathbf{x}_c$ in the current graph, we seek for neighbors in the reference and we attempt to match the observations through a data association routine. If the results of the observation matching are satisfactory, we initialize the position of the expanded node according to the one provided by the matching procedure. This will make the two corresponding observations in the two different maps overlapping, and we add the corresponding edge to $\mathcal{M}$.

Subsequently, we schedule for expansion all the neighbors of $\mathbf{x}_c$ in $\mathcal{G}_m$, and we set their position based on the newly computed position of $\mathbf{x}_c$ and the connecting edge. The pseudo-code for this procedure is shown in Algorithm 1.

Notice that the algorithm is independent from the sensor used. The only requirement is to provide a function *tryMatch(*$\mathbf{n}, \mathbf{x}_c$*)* that attempts to register the observation made from two nodes, and if found, returns the corresponding transformation. In our current implementation the function *tryMatch(*$\mathbf{n}, \mathbf{x}_c$*)* is implemented by a scan-matching routine.

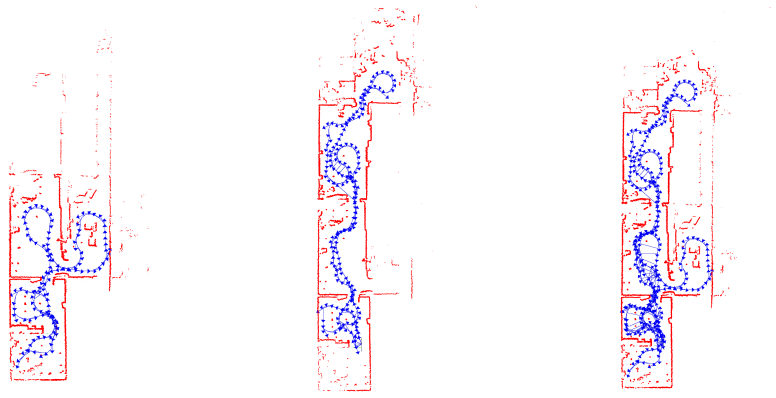
Algorithm 1 Graph Merge

Require: $\mathcal{G}_r$: reference graph, $\mathcal{G}_m$: current graph, $\mathbf{x}_m$: initial vertex in $\mathcal{G}_m$, Δ: minimum distance for matching, *minScore*: minimum score to accept a match.
Ensure: $\mathcal{M}$: edges connecting vertices of $\mathcal{G}_r$ to vertices of $\mathcal{G}_m$.

```
M := ∅; {Initialize the output set of measurements}
x_m.setParent(x_m);
queue.pushBack(x_m);
while ! queue.empty() do
  x_c = queue.front(); {Extract the first node of the matchable graph}
  queue.popFront();
  N = findNeighbors(G_r,x_c); {Find the neighbors of x_c in the reference graph}
  S := ∅; {clear the set of matching results}
  for all n ∈ N do {Try to match each neighbor and put the results in S}
    if |n − x_c| < Δ then
      S.add(tryMatch(n, x_c));
    end if
  end for
  s = bestMatch(S); {Pick up the best match}
  if s.score() > minScore then
    M.add(edge(x_c, n, s.transform())); {Create a new edge and add it to M}
    x_c = n ⊕ s.transform(); {Initialize x_c applying the transform of s to n}
  end if
  for all x_n ∈ neighborsOf(x_c) do
    if x_n.parent() == null; then
      x_n.setParent(x_c);
      e = edgeBetween(x_c, x_n);
      x_n = x_c ⊕ e.transform();
      queue.pushBack(x_n);
    end if
  end for
end while
```

4 Experiments

We evaluated our approach on real and simulated robot systems, and through synthetic experiments. Experiments with raw real data show the applicability of our system to a practical scenario, while synthetic experiments characterize the performance of our method with varying parameters.



(a) First graph (b) Second graph (c) Global map

Fig. 3: This figure illustrates the behavior of our algorithm on the Bremen University dataset. *a), b)* are the input pose graphs, representing different portions of the same environment. *c)* shows the merged pose graph.

4.1 Raw Data Experiments

Our scenario consists in a mobile robot equipped with a laser range finder and a test environment. We acquired the datasets in different, but partially overlapping, portions of the environment. We processed each dataset with a graph-based SLAM engine to obtain the corresponding pose graph. We then compute two solutions, one obtained using our approach, and another derived via a gradient descent algorithm. Finally, the computed solutions are compared evaluating their entropy, [2]. To analyze the behavior of the algorithm when working on grid maps, obtainable using also non graph-based SLAM engines like GMapping [9], we computed an occupancy grid map out of each pose graph. From each grid map, we recompute a pose graph as described in Section 3.2, and we seed them as input to our algorithm.

We conducted the experiments above using a real robot in our building (Dis-Basement). In addition, we used some public datasets [10] to generate realistic maps. We then used the Stage simulator to record multiple datasets of the same environment. Table 1 summarizes the results of our experiments, while Figure 3 shows a typical result. In all cases we analyzed the final solution of our system provided a lower entropy than the baseline, thus a more consistent map.

4.2 Synthetic Experiments

We found that the dominant aspect influencing the behavior of our system is the error affecting the local maps solutions. To quantify this effect, we generated a set of synthetic pose graphs of a robot moving in a Manhattan world. We generated edges between nearby poses and we corrupted them with Gaussian

Dataset	Single Rigid Transformation	Graph-Based Map Merging
Dis-Basement-Small	2039.99	**1538.54**
Dis-Basement-Big-Real	2144.23	**2090.17**
Dis-Basement-Big-Voronoi	4059.91	**3856.92**
Dis-F1-Real	5639.96	**5528.97**
Dis-F1-Voronoi	5928.52	**5778.84**
UBremen-Real	3436.44	**3308.56**

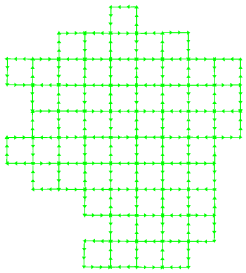
Table 1: Analysis of the entropy of reconstructed global maps. We compare the result of a gradient descent-based technique, second column, against the approach presented in this paper, third column. In bold, we highlight the best result, for each dataset used for the validation.
A more detailed description, together with the datasets, can be found at:
`www.dis.uniroma1.it/~bonanni/datasets`

noise. The noise in the edges models the errors in the matching procedure used by a SLAM algorithm, and affects the final solution. Given two partial pose graphs, we compute the ideal merged graph by adding all the edges between nearby nodes in the two input graphs and we optimize the final result. This result is the best we can do with the noisy input data, and serves us as a baseline. The ideal edges are computed from the ground truth accessible by the simulator. Figure 4 shows the typical pose graphs used in these experiments. In addition to the sensor noise, we also characterized the potential failures of the matching routine used in the *tryMatch()* function of Algorithm 1. The chances of success of a scan matcher depend mostly on how good is the initial guess. We simulated a scan matcher, by implementing a *tryMatch()* function that, having access to the ground truth, reports a solution only if the relative transform of the nodes passed as argument is close to the ground truth. Furthermore, we corrupt the resulting transformations, by adding Gaussian noise.

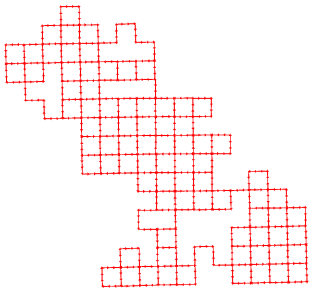
We evaluate the quality of the solution of the graph matching by measuring the *Absolute Trajectory Error* (ATE), [23], between the ideal solution and the one computed by our algorithm. For sake of comparison, we also compute the ATE of the best solution that can be found by a single rigid transformation, by using the *Iterative Closest Point* algorithm, ICP, [17], between corresponding nodes. In all cases the ATE of the merged map using our approach was substantially smaller than the baseline obtained by rigid body transformation. As expected, the quality of the solution decreases as the error residual increases, and the estimate is mostly affected by the rotational part of the error.

5 Conclusion

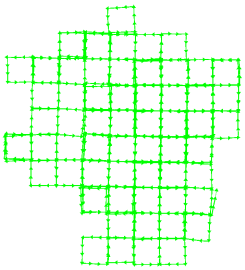
In this paper we presented a generic approach to merge maps described as pose graphs. In case the maps are available as grid maps we provided a technique to extract plausible pose graphs based on Voronoi diagrams. Our method is able to cope with residual errors affecting the input maps, and to remove the artifacts



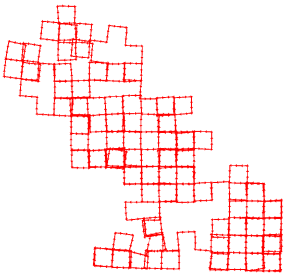
(a) Reference trajectory without noise



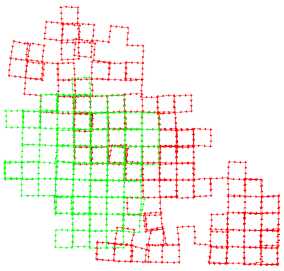
(b) Matchable trajectory without noise



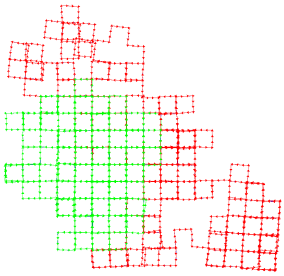
(c) Reference trajectory with noise



(d) Matchable trajectory with noise



(e) Standard map merging approach



(f) Our approach based on deformable bodies

Fig. 4: This figure shows a synthetic experiment where we evaluate our approach on a simulated Manhattan world. *a), b)* show the ground truth trajectories of the robot. *c), d)* show the outcome of a graph-based SLAM algorithm on these trajectories. The residual error results in inconsistencies. *e), f)* show the result of a standard map merging procedure that just overlays the maps with a single transformation and of our approach.

that a standard map merging method leaves in. We validated our approach on real world datasets, and we characterized its sensibility to the noise in the input solutions by simulation, although a more detailed evaluation and a more precise comparison with other existing methods is in progress. Future work is mainly

Translational error [x, y] (m)	Rotational error (deg)	Rigid Transformation ATE	Deformable Bodies ATE
[0.05, 0.01]	2	469.074	**10.2425**
[0.1, 0.01]	2	721.868	**42.5413**
[0.15, 0.01]	2	719.509	**150.515**
[0.2, 0.01]	2	877.672	**243.552**
[0.25, 0.01]	2	989.12	**523.802**
[0.05, 0.02]	2	402.974	**9.49795**
[0.1, 0.02]	2	682.997	**32.4582**
[0.15, 0.02]	2	893.562	**62.5047**
[0.2, 0.02]	2	1029.25	**296.88**
[0.25, 0.02]	2	1130.6	**394.299**
[0.05, 0.03]	2	336.44	**10.4215**
[0.1, 0.03]	2	593.754	**30.4705**
[0.15, 0.03]	2	825.884	**59.5759**
[0.2, 0.03]	2	1058.69	**310.821**
[0.25, 0.03]	2	1329.29	**421.79**

Table 2: Analysis of the absolute trajectory error at increasing levels of gaussian noise. For the translational error, we perturbed both the x, and y axes. In bold, we highlight the best result, at the given noise configuration. Our approach, fourth column, offers better performance with respect to the alignment obtained by a single rigid transformation, third column.

focused at extending the approach to the 3D case, since it represents the new horizon of different modern SLAM techniques.

Acknowledgments

This work has partly been supported by the European Commission under FP7-600890-ROVINA.

References

1. Agarwal, P., Tipaldi, G.D., Spinello, L., Stachniss, C., Burgard, W.: Robust map optimization using dynamic covariance scaling. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). (May 2013)
2. Blanco, J.L., Fernández-Madrigal, J.A., Gonzalez, J.: An entropy-based measurement of certainty in rao-blackwellized particle filter mapping. In: Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on, IEEE (2006) 3550–3555
3. Bunke, H.: Graph matching: Theoretical foundations, algorithms, and applications. In: Proc. Vision Interface. Volume 2000. (2000) 82–88
4. Carlone, L., Ng, M.K., Du, J., Bona, B., Indri, M.: Rao-blackwellized particle filters multi robot slam with unknown initial correspondences and limited communication. In: Robotics and Automation (ICRA), 2010 IEEE International Conference on, IEEE (2010) 243–249

5. Carpin, S.: Fast and accurate map merging for multi-robot systems. Autonomous Robots **25**(3) (2008) 305–316
6. Cunningham, A., Paluri, M., Dellaert, F.: Ddf-sam: Fully distributed slam using constrained factor graphs. In: Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on, IEEE (2010) 3025–3030
7. Erinc, G., Carpin, S.: Anytime merging of appearance-based maps. Autonomous Robots **36**(3) (2014) 241–256
8. Grisetti, G., Kümmerle, R., Stachniss, C., Burgard, W.: A tutorial on graph-based slam. Magazine on Intelligent Transportation Systems **2**(4) (2010) 31 – 43
9. Grisetti, G., Stachniss, C., Burgard, W.: Improved techniques for grid mapping with rao-blackwellized particle filters. IEEE Trans. on Robotics **23**(1) (2007) 34–46
10. Howard, A., Roy, N.: The robotics data set repository (Radish) (2003) http://radish.sourceforge.net/.
11. Huang, W.H., Beevers, K.R.: Topological map merging. The International Journal of Robotics Research **24**(8) (2005) 601–613
12. Jennings, J., Kirkwood-Watts, C., Tanis, C.: Distributed map-making and navigation in dynamic environments. In: Intelligent Robots and Systems, 1998. Proceedings., 1998 IEEE/RSJ International Conference on. Volume 3., IEEE (1998) 1695–1701
13. Kümmerle, R., Grisetti, G., Strasdat, H., Konolige, K., Burgard, W.: g2o: A general framework for graph optimization. In: Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA). (2011)
14. Lazaro, M., Paz, L., Piniés, P., Castellanos, J., Grisetti, G.: Multi-robot slam using condensed measurements. In: Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on, IEEE (2013) 1069–1076
15. Leung, K.Y.K., Barfoot, T.D., Liu, H.H.: Distributed and decentralized cooperative simultaneous localization and mapping for dynamic and sparse robot networks. In: Robotics and Automation (ICRA), 2011 IEEE International Conference on, IEEE (2011) 3841–3847
16. Lu, F., Milios, E.: Globally consistent range scan alignment for environment mapping. Autonomous Robots **4** (1997) 333–349
17. Lu, F., Milios, E.: Robot pose estimation in unknown environments by matching 2d range scans. Journal of Intelligent and Robotic Systems **18**(3) (1997) 249–275
18. Montemerlo, M., Thrun, S., Koller, D., Wegbreit, B.: FastSLAM: A factored solution to simultaneous localization and mapping. In: Proc. of the National Conference on Artificial Intelligence (AAAI), Edmonton, Canada (2002) 593–598
19. Olson, E., Leonard, J., Teller, S.: Fast iterative optimization of pose graphs with poor initial estimates. In: Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA). (2006) 2262–2269
20. Saeedi, S., Paull, L., Trentini, M., Seto, M., Li, H.: Map merging using hough peak matching. In: Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on, IEEE (2012) 4683–4688
21. Smith, R., Self, M., Cheeseman, P.: Estimating uncertain spatial realtionships in robotics. In Cox, I., Wilfong, G., eds.: Autonomous Robot Vehicles. Springer Verlag (1990) 167–193
22. Thrun, S., Liu, Y., Koller, D., Ng, A., Ghahramani, Z., Durrant-Whyte, H.: Simultaneous localization and mapping with sparse extended information filters. Int. Journal of Robotics Research **23**(7/8) (2004) 693–716
23. Wulf, O., Nuchter, A., Hertzberg, J., Wagner, B.: Ground truth evaluation of large urban 6d slam. In: Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on, IEEE (2007) 650–657