# Dynamic Pose Graph SLAM: Long-term Mapping in Low Dynamic Environments

Aisha Walcott-Bryant, Michael Kaess, Hordur Johannsson, and John J. Leonard

***Abstract*— Maintaining a map of an environment that changes over time is a critical challenge in the development of persistently autonomous mobile robots. Many previous approaches to mapping assume a static world. In this work we incorporate the time dimension into the mapping process to enable a robot to maintain an accurate map while operating in dynamical environments. This paper presents Dynamic Pose Graph SLAM (DPG-SLAM), an algorithm designed to enable a robot to remain localized in an environment that changes substantially over time. Using incremental smoothing and mapping (iSAM) as the underlying SLAM state estimation engine, the Dynamic Pose Graph evolves over time as the robot explores new places and revisits previously mapped areas. The approach has been implemented for planar indoor environments, using laser scan matching to derive constraints for SLAM state estimation. Laser scans for the same portion of the environment at different times are compared to perform change detection; when sufficient change has occurred in a location, the dynamic pose graph is edited to remove old poses and scans that no longer match the current state of the world. Experimental results are shown for two real-world dynamic indoor laser data sets, demonstrating the ability to maintain an up-to-date map despite long-term environmental changes.**

## I. Introduction

One of the long-term goals in mobile robotics is to achieve life-long mapping — the ability to construct and maintain an up-to-date map while operating persistently in an environment that undergoes substantial changes over time. We refer to this as the long-term simultaneous localization and mapping (SLAM) problem in dynamic environments. A robot that repeatedly traverses a typical dynamic environment will encounter objects that move at wide-varying time scales. In this work we focus on the robot mapping problem in *low-dynamic* environments. Low dynamic environments are composed of static and low-dynamic objects and entities that can be moved or changed at any time, such as furniture or walls in a remodeled room. The dynamics of such an environment directly affect what the robot senses and ultimately incorporates into its map. In this paper, we present the Dynamic Pose Graph (DPG) model to represent changing environments, and the DPG-SLAM method for continuous long-term mapping in low-dynamic environments. The aim of our work is to maintain a current up-to-date map at all times, identifying the more static components of the world, to facilitate localization and to provide a foundation for future work in active exploration that would account for long-term environment dynamics.

Recently, some successful approaches to dealing with low-dynamic objects have been proposed, however there remain a number of open challenges. Andrade-Cetto [1] presents an early system for landmark-based mapping in dynamic environments using an extended Kalman filter. Wolf and Sukhatme [2] and Biswas et al. [3] instead use occupancy grids to maintain both a static and a dynamic map of a changing environment. Mitsou and Tzafestas [4] modify the occupancy grid structure to maintain a history of sensor values. Performing SLAM with occupancy grids suffers from deteriorating map quality over time. More recently pose graph optimization techniques [5] have been very successful at constructing accurate maps by correcting older robot poses with, for example, loop closures.

Additionally, one of the key questions in long-term mapping is: what should be used to represent the map of a changing environment? Biber and Duckett [6], [7] presented a sample based representation that employs multiple maps, each adapting to environment changes at a different time scale. Konolige and Bowman [8] discuss a visual mapping system that attempts to repair the map as the environment changes. Meyer-Delius [9] developed a dynamic occupancy grid that models temporal changes in the environment. Wyeth et al. [10] take a biologically inspired approach, called RatSLAM, that has been tested in the context of an office delivery robot, using a non-metric representation.

Our proposed approach is based on state-of-the-art pose graph optimization techniques for SLAM. In general, pose graphs grow with time, are sensitive to noisy constraints, and can become highly connected when a robot repeatedly revisits an area and loop constraints are added. Our objective is to continuously maintain a pose graph representation over time, keeping a history of the environment dynamics.

The primary contribution of this work is the introduction of a novel model called the Dynamic Pose Graph (DPG). Pose graphs optimize the robot poses explicitly, while the map for a static environment is implicitly represented by the associated laser scans. The DPG extends this model for low-dynamic environments by maintaining both an active and a dynamic map. By annotating the scans, or subsets of scans, for finer granularity, dynamic objects can be isolated, and the most recent map recovered at any time. And by keeping the history in the pose graph, a more accurate and complete model can be achieved than by only using the latest data.

This research has been supported by the Lucent CRFP, the MIT GSO-Lemelson Fellowship, and the MIT Graduate Students Office Fellowship. Also, this work is partially the result of research sponsored by the MIT Sea Grant College Program, under NOAA Grant Number NA06OAR4170019, MIT SG Project Number 2007-R/RCM-20, and by the Office of Naval Research under research grants N00014-10-1-0936 and N00014-12-10020.

The authors are with the Massachusetts Institute of Technology, Cambridge, MA 02139, USA. aisha.w.bryant@gmail.com, {kaess, hordurj, jleonard}@mit.edu

The second contribution is Dynamic Pose Graph SLAM (DPG-SLAM), which adds complexity management to directly address the problem of long-term mobile robot navigation and map maintenance in dynamic environments. By carefully selecting nodes and constraints that are no longer needed (due to changes in the world), the complexity of the pose graph can be substantially reduced.

We demonstrate our work on two real-world dynamic indoor laser data sets collected by a B21 mobile robot. Our results demonstrate DPG-SLAM's ability to maintain an efficient, up-to-date map despite long-term environmental changes.

The structure of this paper is as follows: Section II describes the Dynamic Pose Graph representation. Section III provides the overall DPG-SLAM algorithm. Section IV presents our experimental results, and finally Section V concludes the paper by summarizing our contributions and making suggestions for future research.

## II. Dynamic Pose Graph (DPG)

In this section, we describe our environment model and assumptions, and present an example of a low-dynamic environment. Then we introduce the Dynamic Pose Graph (DPG) model.

### A. Environment Model and Assumptions

A general dynamic environment model captures moving, low-dynamic, high-dynamic, and stationary objects, in addition to entities (such as walls or other physical structures) that can change. Non-stationary objects move at wide-varying time scales from seconds, such as people walking, to weeks, such as a piece of furniture moved from one location to another. To construct a map, we assume that high-dynamic objects can be filtered by employing existing methods including [11]–[13]. Thus, in this work we focus on mapping low-dynamic environments. More specifically, a low-dynamic object moves much slower than the robot navigates and its motion cannot be sensed immediately.

An example of an indoor low-dynamic environment, the Reading Room, where boxes are intentionally moved is shown in Fig. 1. The Figure shows four separate pose graph SLAM laser range maps created after the robot traverses the environment four times.

We make some assumptions about the environment that allow us to focus on the core issues of the dynamic mapping problem. We assume that the space is a bounded two-dimensional indoor, office environment. The robot performs multiple passes through the same environment over time, where each pass starts and ends near a known home location. Changes in the environment are a result of (1) low-dynamic objects being added, removed, or moved, or (2) changes in presumed static objects (eg. walls during remodeling). We also assume that if changes occur, then they occur between passes.

One of the goals of the DPG-SLAM method is to identify stale incorrect map measurements, as shown in Fig. 1(b), and remove it from the representation. Applying pose graph SLAM results in a cluttered map with stale data. A second goal of the DPG-SLAM method is to create a map representation that incorporates both the static and the dynamic parts of the environment. Fig. 1(c) depicts an example of the best-case DPG-SLAM map, called *active map*, for the Reading Room. The map contains all the static parts (walls) from each pass, and removes the portions of the map that are incorrect, as a result of boxes moved at different points in time. Correspondingly, Fig. 1(d) depicts the best-case DPG-SLAM map of the dynamic parts of the environment, called the *dynamic map*. This map shows the history where boxes were added, moved, or removed. By maintaining the static parts and labeling the dynamic parts of the environment, the areas that are more static can later be used for reliable localization.

### B. The Dynamic Pose Graph Model

We present a novel model called the Dynamic Pose Graph (DPG) to address the problem of long-term mapping in dynamic environments. The DPG is an extension of the traditional pose graph model. A Dynamic Pose Graph is a connected graph, denoted ***DPG*** = $\langle N, E\rangle$, with nodes $n_i \in N$ and edges $e_{i,j} \in E$ and is defined as follows:

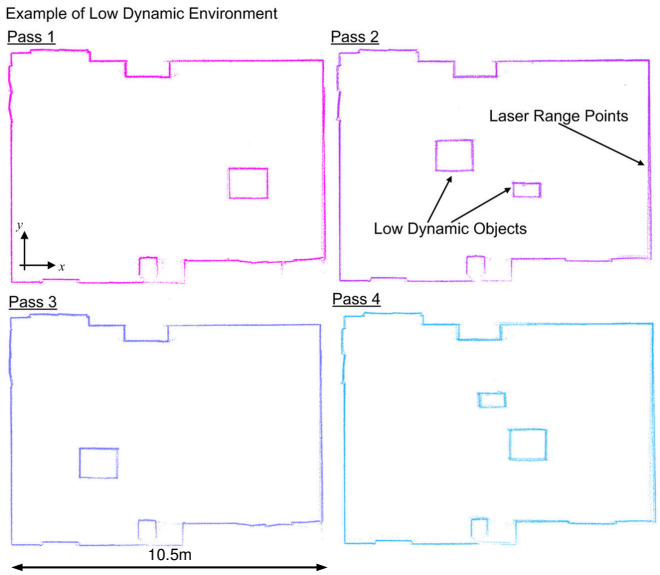
**Dynamic Pose Graph, *DPG* =** $\langle N, E\rangle$:

- *Node* $n_i \in N$, where $n_i = \langle \mathbf{x}, c, a, p, z\rangle$ with
  - *Pose* $\mathbf{x} = \begin{bmatrix} x_i & y_i & \theta_i \end{bmatrix}^T$
  - *Change node indicator*
    $$c = \begin{cases} 1, & \text{if change detected} \\ 0, & \text{otherwise} \end{cases}$$
  - *Active node indicator*
    $$a = \begin{cases} active, & \text{if measurement is used in the map} \\ inactive, & \text{otherwise} \end{cases}$$
  - *Pass number* $p$, an integer representing the pass at which the node was created
  - *Measurement* $z$ taken at node $n_i$.
- *Edge* $e_{i,j} \in E$, where $e_{i,j} = \langle T, \Sigma\rangle$ with
  - *Constraint* $T_{i,j}$ is determined by computing spatial constraints between two nodes $n_i$ and $n_j$, where the spatial geometric transform is $T_{i,j} = \begin{bmatrix} x_{ij} & y_{ij} & \theta_{ij} \end{bmatrix}^T$
  - *Covariance* $\Sigma \in \mathbb{R}^{3\times 3}$ of the constraint $T_{i,j}$

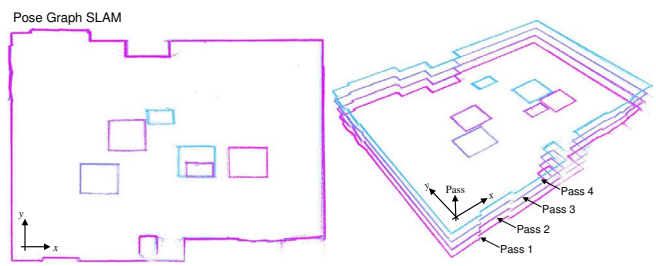
An example of a Dynamic Pose Graph is provided in Fig. 2. There, the DPG is shown originating from the known start position at each pass.

### C. Long-term Map Representation

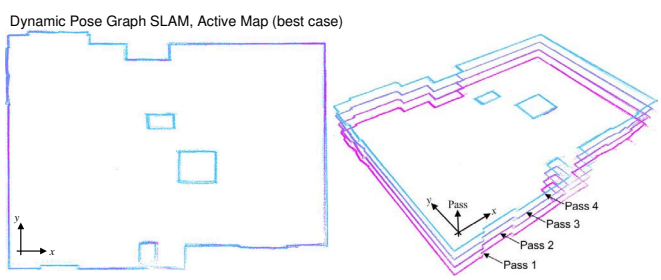
The map representation in the DPG is similar to that of a standard pose graph. In the standard pose graph a map can be generated by projecting the measurements (e.g. laser scans) taken at each pose into a single global coordinate frame. As a result, the map $Z$ can be represented by the set of all measurements, $z_i \in Z$. This section describes additional information stored at each DPG node, as well as the active and dynamic maps.



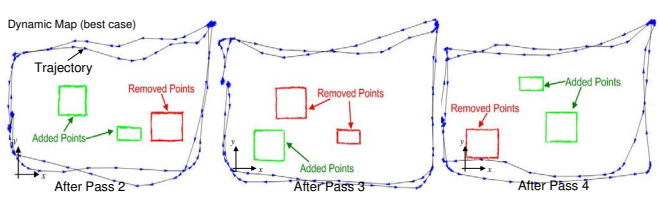

(a) Example of a low-dynamic environment with range scans generated from the CSAIL Reading Room. Low-dynamic objects are boxes that are intentionally added, moved, or removed after each pass.




(b) Pose graph SLAM map.




(c) DPG-SLAM best-case active map.




(d) DPG-SLAM best-case dynamic map.

Fig. 1. (a) Example of a low-dynamic environment with range scans generated from the CSAIL Reading Room. Low-dynamic objects are boxes that are intentionally added, moved, or removed after each pass. (b) Pose graph SLAM map. (c) Best-case active map. (d) Best-case dynamic map.

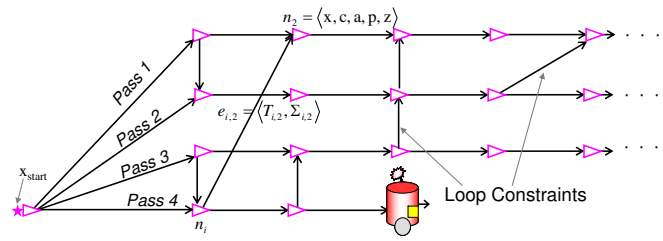



Fig. 2. Example of a Dynamic Pose Graph (DPG).

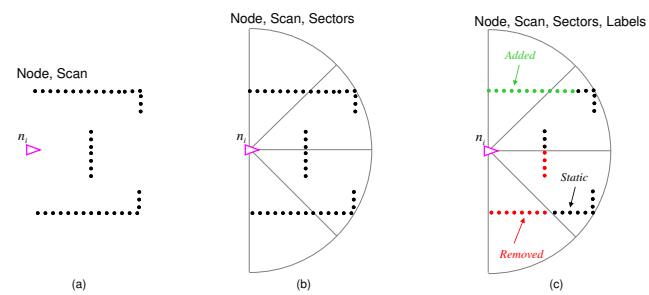



Fig. 3. Illustration of a DPG node and its contents. (a) Shows a standard pose graph node. (b) Sectors are added. (c) Complete DPG node with four sectors and laser points that are labeled static, added, or removed.

*1) Nodes and Measurements:* We use laser range scans as measurements, and construct a map using partial scans and labels. More specifically, a measurement $z$ is a *laser range scan*, where $z = \langle \psi, \alpha \rangle$,

- *Range values and labels* $\psi = \{\langle r_1, l_1 \rangle ..., \langle r_m, l_m \rangle\}$, where $m$ is the number of ranges in the laser scan. *Range labels* $l_k = \{static, added, removed\}$ denote which laser range measurements are derived from a low-dynamic object or a static object. A low-dynamic object is either added, removed, or a combination thereof.
- *Sectors* $\alpha$ divide the laser scan into equal-sized partitions based on angle. The purpose of sectors is to retain as many accurate ranges as possible by dividing the laser scan and removing only the parts of the scan that should not be included in the map. Each laser scan has a set of $b$ sectors, $A = \{ \alpha_1, \alpha_2, ..., \alpha_b \}$.

An example of a DPG node is shown in Fig. 3, where (a) shows a node from a standard pose graph, and (b) and (c) show additional components, sectors and labels, included in a DPG node.

*2) Active and Dynamic Maps:* To represent a changing environment, the DPG maintains two maps, an *active map* and a *dynamic map*. The active map represents the most current state of the environment including parts of the environment that have not changed from previous passes. An active map is defined as follows. **Active Map:** $Z_{active}$ = $\{ r_i \mid r_i$ is a range point from scan $z_j$, and $\alpha(r_i)$ = on, and range label $l_i \in \{static, added\}$ and node($z_j$) = active $\}$. $Z_{active}$ is the set of range measurements from laser scans that correspond to active nodes; the corresponding sector of each range measurement must be *on*, $\alpha(r_i) = on$; the label

for each range is either static or added.

The dynamic map contains a representative sample of laser range points from parts of the environment that have changed over time, and is defined as follows. **Dynamic Map:** $Z_{dynamic} = \{ r_i \mid r_i$ is a range point from scan $z_j$, with range label $l_i \in \{removed, added\} \}$. $Z_{dynamic}$ is the set of ranges deriving from active or inactive nodes and are labeled either added or removed. Note that added points are also included in the dynamic map because they represent a change in the environment. The dynamic map is a representation of the history of changes that have been detected.

## III. DPG-SLAM

For a mobile robot to be able to navigate in a dynamic environment and maintain an up-to-date map, it must be able to continuously localize, detect changes, and repair its map. To achieve this, we present DPG-SLAM a method to maintain an up-to-date representation while a robot navigates in a low-dynamic environment. DPG-SLAM addresses two key challenges of the long-term mobile robot mapping problem. The first challenge is to maintain a map of a low-dynamic environment that changes over time. The second challenge is to reduce the size of the DPG as it grows over time. The DPG-SLAM steps are as follows,

**DPG-SLAM Steps:**

1) Pose graph SLAM
2) Compute local submap
3) Detect and label changes
4) Update active and dynamic maps
5) Reduce DPG size
6) Repeat

DPG-SLAM addresses the challenges with the DPG-SLAM-NR algorithm, where DPG-SLAM-NR (No Reduce/Remove) excludes step 5. The nodes and edges in DPG-SLAM-NR refer to the robot's entire trajectory, $X$; the nodes and edges in DPG-SLAM can be removed during step 5, and thus refer to only a subset of the robot's trajectory, $X^* \subseteq X$.

### A. Compute Local Submap

To create an accurate map of a changing environment, the robot must be able to detect changes and update its active and dynamic maps. That is, the robot must be able to continuously compare the current state of the environment to its previous representation. To achieve this, we introduce two terms: *current pose chain* and *local submap* (see Fig. 4), whose sets of range points are used in change detection. The current pose chain is the most recent sequence of nodes added to the DPG. A local submap is a subset of active map nodes and points from earlier passes. Each node in the local submap has a field-of-view (FOV), free unobstructed space measured by the laser range scanner, that intersects with at least one pose chain node's FOV, or vice versa.

To compute the local submap we find a subset of active map nodes that sufficiently cover, based on intersecting FOVs, the pose chain nodes. This subset of nodes are in close proximity to the pose chain nodes. The nodes also must pass a $\chi^2$ test for the relative uncertainty (covariance)

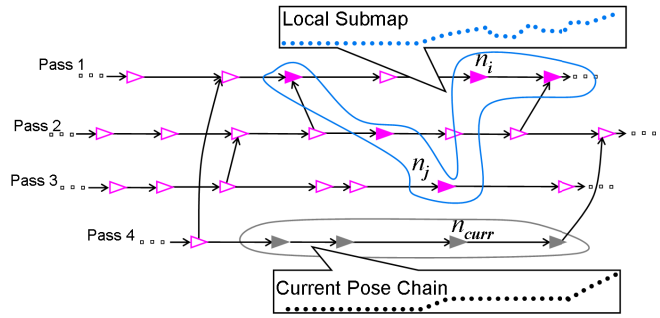



Fig. 4. Example of the current pose chain and a local submap.

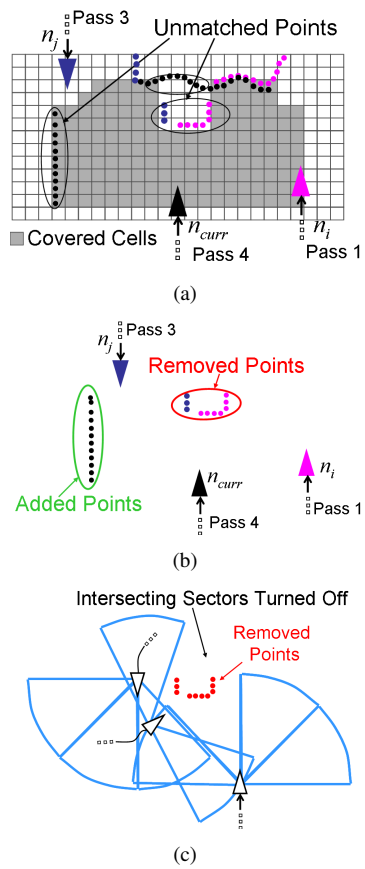



Fig. 5. (a) Example of pose chain node $n_{curr}$'s cells covered by local submap nodes $n_i$ and $n_j$, and their identified unmatched points. (b) Labeled added and removed points. (c) Example of sectors that intersect with removed points turned off.

between each node and at least one pose chain node [14]–[17]. We determine sufficient coverage by overlaying two occupancy grids: one occupancy grid for the local submap, $submapGrid$, and one for the current pose chain $currGrid$. Then if the amount of covered cells from the $currGrid$ exceeds a given threshold, the pose chain is sufficiently covered by the local submap. An example of finding local submap nodes for a pose chain node, $n_{curr}$, is shown in Fig. 5(a). The figure highlights the covered area w.r.t. $n_{curr}$ and two local submap nodes, $n_i$ and $n_j$.

TABLE I

RULES FOR LABELING POINTS.

| $submapGrid$ | $currGrid$ | Label |
|---|---|---|
| free | occupied | current points *added* |
| occupied | free | submap points *removed* |
| unknown | occupied | current points *static* |

### B. Detect and Label Changes

Recall that low-dynamic objects can be added, moved, or removed after each pass. To detect and label changes resulting from low-dynamic objects, we apply three steps to each node, $n_{curr}$ on the pose chain.

The first step is to compute the set of unmatched points, denoted $\eta$, consisting of unmatched points from $n_{curr}$ and unmatched points from $\gamma$, where $\gamma$ is the set of local submap nodes that intersect with $n_{curr}$. To compute $\eta$, scan matching is applied to find the points that are not matched (unmatched points). An alternative method is to use an occupancy grid technique to differentiate between static and dynamic elements as in [3].

The second step is to compute a change score to determine if the amount of change exceeds a given threshold, $\delta$. We use the following function to determine if a change has occurred,

$$c_{curr} = \begin{cases} 1, & \textit{if score}(z_{curr},\ \eta) > \delta \\ 0, & \textit{otherwise}, \end{cases}$$

To compute the *score*, $n_{curr}$'s sensor range is temporarily divided into equal sized bins by angle. Each unmatched point is assigned to a bin based on its angle relative to $n_{curr}$. The score is computed from the ratio of the number of bins containing unmatched points to the remaining bins. If the score exceeds $\delta$ then change(s) has been detected.

**Alg. 1** LABEL-POINTS(OccGrid $currGrid$, OccGrid $submapGrid$)

$dynMapPoints \leftarrow \{\}$
**for all** $cell \in currGrid$ **do**
  **if** ($cell$ == occ) AND ($submapGrid[cell]$ == free) **then**
    **for all** $p_i \in cell.points$ **do**
      label($p_i$) $\leftarrow$ added
      $dynMapPoints \leftarrow dynMapPoints \cup p_i$
    **end for**
  **else if** ($cell$ == free) AND ($submapGrid[cell]$ == occ) **then**
    **for all** $p_j \in submapGrid[cell].points$ **do**
      label($p_j$) $\leftarrow$ removed point
      $dynMapPoints \leftarrow dynMapPoints \cup p_j$
    **end for**
  **end if**
**end for**
return *dynMapPoints*

The third step is to label the points from $z_{curr}$ and $\gamma$ as *static*, *removed*, or *added* to update the dynamic map. The procedure for labeling points is given in Alg. 1, and an example is shown in Fig. 5(b). The occupancy grid for the current node, $currGrid$, and the occupancy grid for the local submap nodes, $submapGrid$, are overlaid. Cells of each of the two grids are compared, and points in these cells are labeled according to the rules in Table I, later used to update the dynamic map.

### C. Update Active and Dynamic Maps

To update the dynamic and active maps, the labeled pose chain and local submap points. The dynamic map is updated to include the added and removed points. The active map is updated to include the added and static points. This procedure is given in Alg. 2. Additionally, sectors of DPG nodes in the active map are "turned off" if the sectors intersect removed points, i.e. stale data, see Fig. 5(c). Points in sectors that are turned off are no longer included in the active map.

**Alg. 2** UPDATE-MAPS(DPG $dpg$, Points $removedPoints$)

$inactiveNodes \leftarrow \{\}$
**for all** $p_i \in removedPoints$ **do**
  $nearbyNodes \leftarrow$ get_potential_overlapping_nodes($p_i$, $dpg$)
  **for all** $n_j \in nearbyNodes$ **do**
    **if** active($n_j$) == true **then**
      **if** intersects($p_i$, $n_j$) **then**
        $sector \leftarrow$ get_sector($n_j$, $p_i$)
        set_state($sector$) $\leftarrow$ Off
        **if** percent on-sectors $< percOn$ **then**
          set_state($n_j$) $\leftarrow$ inactive
          $inactiveNodes \leftarrow inactiveNodes \cup n_j$
        **end if**
      **end if**
    **end if**
  **end for**
**end for**
return $inactiveNodes$

### D. Removing DPG Nodes and Constraints

As the robot navigates the DPG will continue to grow. As a result, pose graph optimization becomes more and more computationally expensive. To address the DPG size we attempt to remove the *inactive* nodes from the graph. An inactive node is a node with all sectors turned off. Removing nodes together with their corresponding laser scans reduces the set of nodes (or poses) $X$, to a subset $X^* \subseteq X$.

**Alg. 3** REMOVE-INACTIVE-NODES(DPG $dpg$)

**for all** $n_i \in$ inactive_ nodes($dpg$) **do**
  Traverse back from $n_i$ to nodes from same pass as $n_i$
  Insert start constraint
  Traverse forward from $n_i$ to nodes from same pass as $n_i$
  Insert end constraint
  Create removal chain
**end for**
Remove removal chain if valid

An illustration of removing an inactive node is shown in Fig. 6, and detailed in Alg. 3. To remove inactive nodes there are four issues to consider. The first is that DPG must remain connected at all times. The second consideration is that additional nodes will likely be removed with an inactive node (termed a removal chain in Alg. 3). The third is that removing nodes requires removing constraints which at times may greatly affect the pose estimates. The fourth consideration is that the number of active nodes removed from the DPG affect the active map coverage.

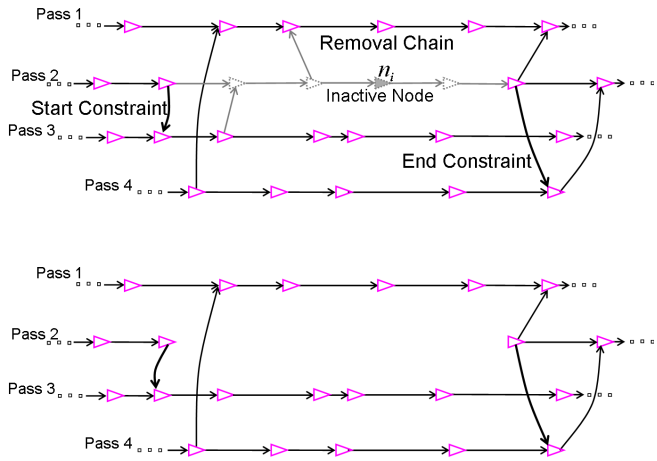



Fig. 6. Removing an inactive node and its corresponding removal chain in order to reduce the size of the DPG. (top) Before removal. (bottom) Reduced DPG.

TABLE II

SUMMARY OF DPG-SLAM PARAMETERS.

| Parameter | CSAIL Reading Room | Univ. of Tubingen |
|---|---|---|
| $\delta$, change ratio | 0.2 | 0.3 |
| Sectors/node | 5 | 8 |
| Removal chain len | 5 nodes | 15 nodes |
| # Passes | 20 | 60 |
| Distance travelled | 1.0km | 7.4km |

## IV. RESULTS

To demonstrate the efficacy of DPG-SLAM and DPG-SLAM-NR we present experimental results for two real-world dynamic indoor laser data sets: the CSAIL Reading Room and the Univ. of Tubingen [6]. Table II summarizes the DPG-SLAM parameters used for each of the data sets. Incremental smoothing and mapping (iSAM) [18] was used for pose graph optimization.

### A. CSAIL Reading Room

In this experiment boxes were added, moved, and removed by hand in order to know where changes occurred. The robot made 20 passes through the room traveling a total distance of 1.0km. Fig. 7 shows the map with all information from applying pose graph SLAM. Fig. 9 shows the maps resulting from applying DPG-SLAM (top) and DPG-SLAM-NR (bottom). A summary of the resulting DPG size is shown in Table III.

To show the accuracy and computation time of the algorithms, we use three metrics: static histograms, ground truth error, and the run-time ratio. Fig. 8 shows the density

TABLE III

DPG SUMMARY FOR CSAIL READING ROOM.

| Algorithm | Nodes | Edges | Rem. Nodes | Rem. Edges | Error ($cm$) |
|---|---|---|---|---|---|
| DPG-SLAM-NR | 2,468 | 3,158 | —- | —- | 3.9-13.4 |
| DPG-SLAM | 1,345 | 1,647 | 1,123 | 1,932 | 3.4-13.3 |

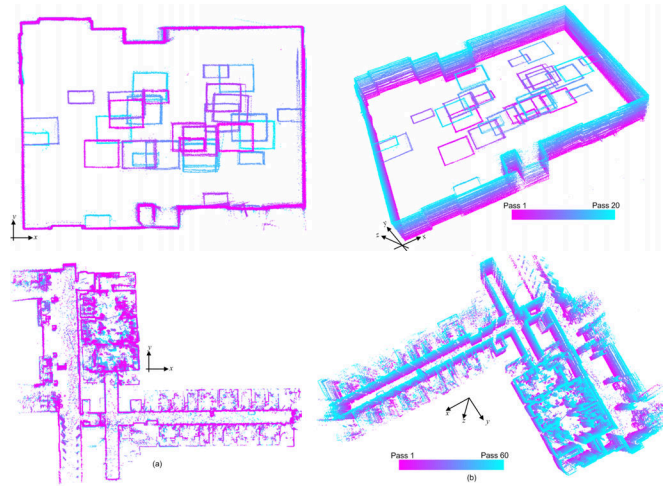



Fig. 7. Top-down and side-views of pose graph SLAM applied to the Reading Room (top) and Univ. of Tubingen (bottom) data sets.

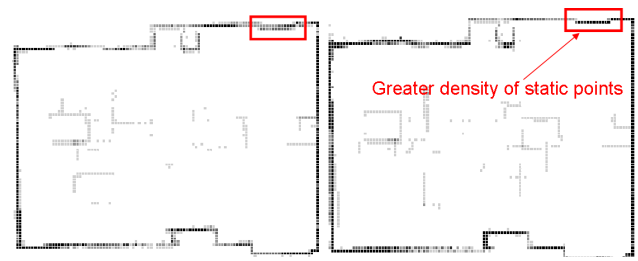



Fig. 8. Static histograms created from Reading Room active maps for DPG-SLAM (left) and DPG-SLAM-NR (right).

of static points (from the walls) is greater for the DPG-SLAM-NR than the DPG-SLAM. This is due to nodes being removed during DPG-SLAM that are not inactive, and thus, their valid laser range points are removed from the active map. Fig. 11 shows the ground truth error—the mean of the distances between active map static points and ground truth, as the number of passes increases. Finally, the run-time ratio between the two algorithms is shown in Fig. 12. During the early iterations the computational cost of DPG-SLAM is slightly greater than DPG-SLAM-NR. However, in later iterations it is clear that reducing the size of the DPG yields significant savings in computation.

### B. University of Tubingen

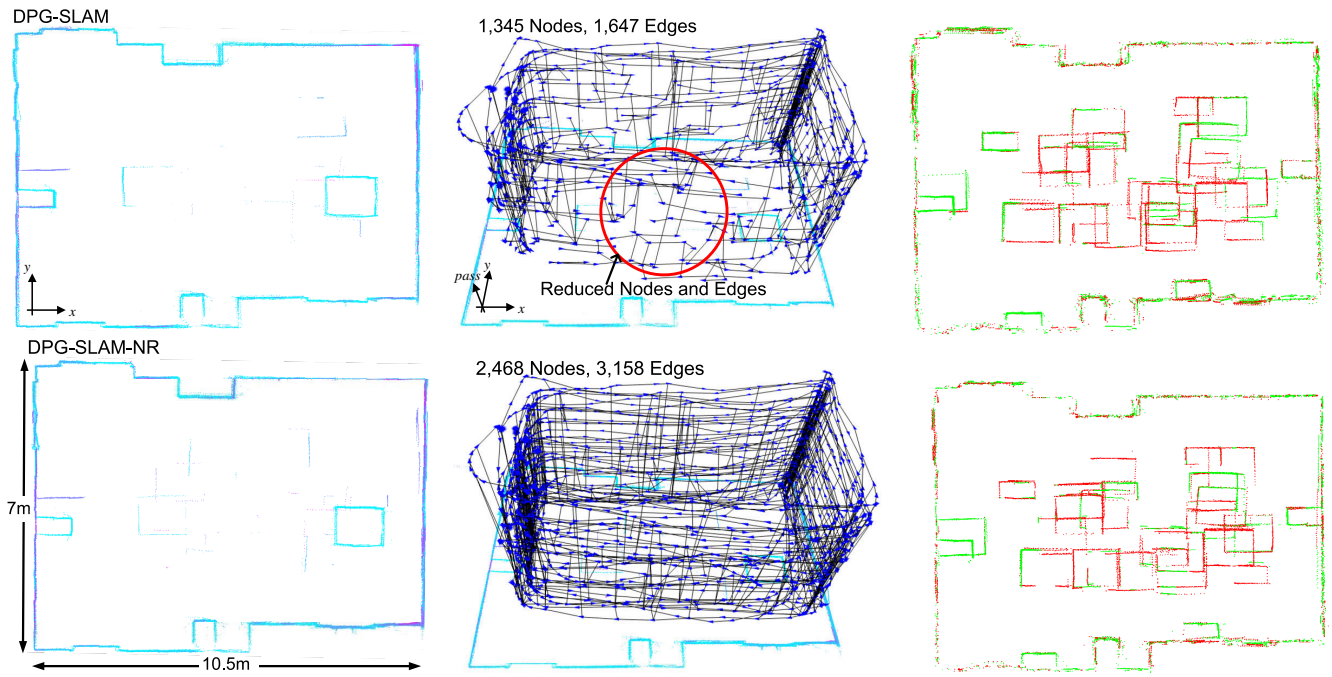
The Univ. of Tubingen data set consists of laser data collected over five weeks in an indoor environment, courtesy of Biber and Duckett [6]. The robot made 60 passes through the building, traveling a total distance of 7.4km.

Fig. 10 shows the maps resulting from applying DPG-SLAM (top) and DPG-SLAM-NR (bottom). The parts of

TABLE IV

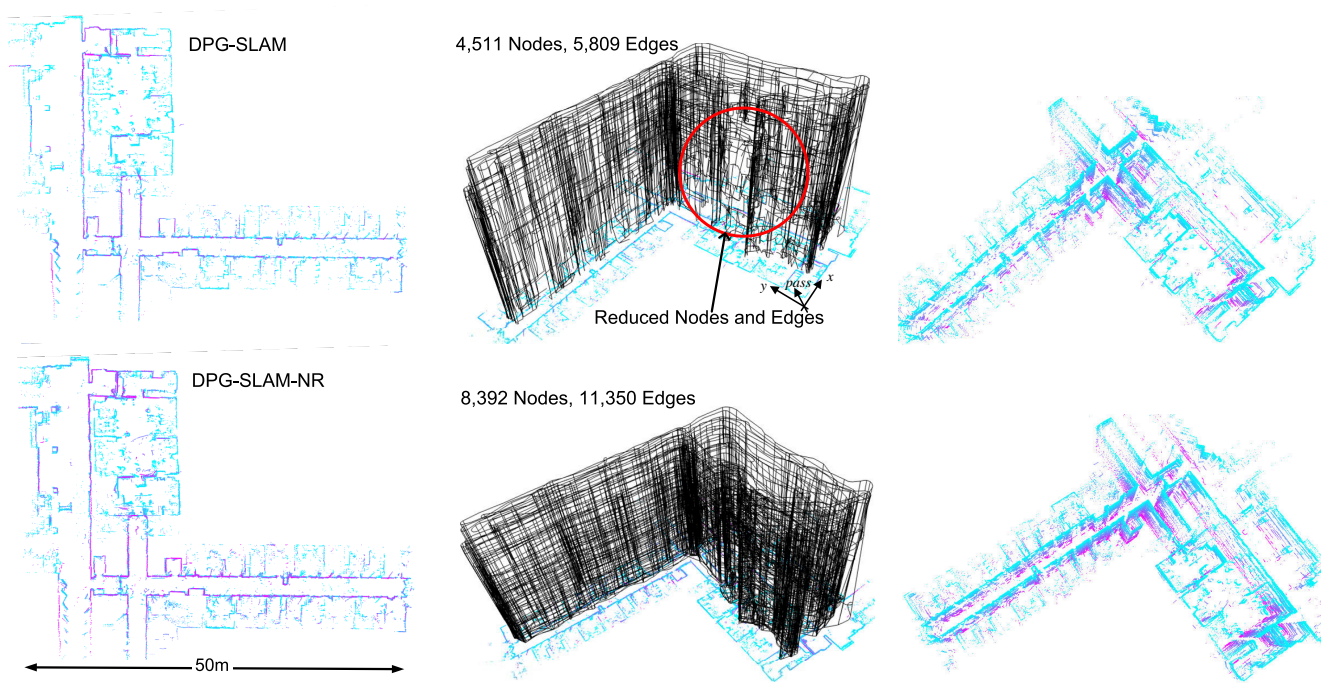
DPG SUMMARY FOR THE UNIV. OF TUBINGEN.

| Algorithm | Nodes | Edges | Rem. Nodes | Rem. Edges |
|---|---|---|---|---|
| DPG-SLAM-NR | 8,392 | 11,350 | —- | —- |
| DPG-SLAM | 4,511 | 5,809 | 3,881 | 7,324 |




(a) Active map (b) DPG (c) Dynamic map

Fig. 9. Results of the DPG-SLAM (top row) and DPG-SLAM-NR (bottom row) algorithms for the Reading Room data set. The points are color-coded from each pass, where magenta is the oldest pass and cyan is the most recent pass.




(a) Active map (b) DPG (c) Side-view Active Map

Fig. 10. Results of the DPG-SLAM (top row) and DPG-SLAM-NR (bottom row) algorithms for the Univ. of Tubingen dataset.

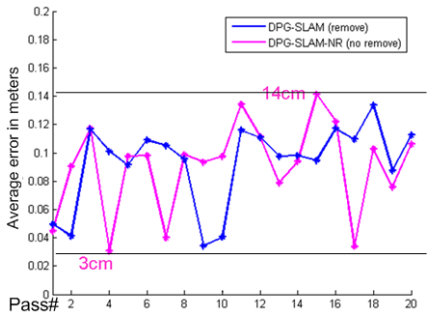



Fig. 11. Plot of errors in ground truth accuracy and DPG-SLAM-NR.

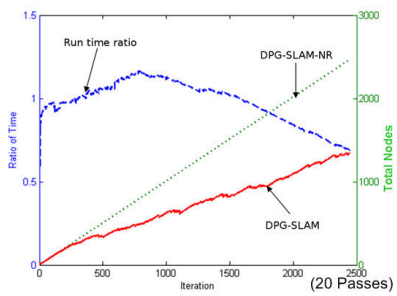



Fig. 12. Run time ratio between both algorithms, as well as the number of nodes added with each algorithm over time.

the active maps shown in magenta imply that data from earlier passes remain in the active map, as shown along the "T" intersection in the middle. Again, the density of static points from the walls is greater for the DPG-SLAM-NR than the DPG-SLAM. Table IV summarizes the DPGs from the two algorithms. The size of the DPG is significantly reduced for the DPG-SLAM algorithm. The active maps from both algorithms are very similar even though the size of the DPG for the DPG-SLAM algorithm is nearly half the size of the one from DPG-SLAM-NR.

## V. Conclusion

To address the problem of long-term mobile robot mapping in low-dynamic environments, we presented the DPG model and DPG-SLAM. The DPG-SLAM-NR algorithm addressed the map maintenance problem in low-dynamic environments, and the DPG-SLAM algorithm extended DPG-SLAM-NR to address the growth in the DPG. Our experiments demonstrate that an accurate map of a changing environment can be maintained, while reducing the size of the DPG and addressing the issue of tractability. There was minimal trade-off in accuracy between the two algorithms, with a great benefit of computation time for DPG-SLAM.

The DPG-SLAM method can be improved in a number of ways. In some cases, the removal of constraints from the graph can affect the quality of the pose estimates. In addition, false negatives (in which stale range points remain in the active map) and false positives (in which range points for stationary objects are erroneously removed) can occur. It is anticipated that with sufficient repeated traversals of an environment, these effects can be minimized.

Key topics to investigate in future work include: (1) extension to 3D mapping using visual SLAM [19]; (2) incorporation of semantic representations for places and objects; and (3) development of motion planning techniques that balance exploration of new areas with map maintenance for previously mapped regions.

## References


[1] J. Andrade-Cetto and A. Sanfeliu, "Concurrent map building and localization on indoor dynamic environments," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 16, no. 3, pp. 361–374, 2002.

[2] D. F. Wolf and G. S. Sukhatme, "Towards mapping dynamic environments," *In Proceedings of the International Conference on Advanced Robotics (ICAR)*, pp. 594–600, 2003.

[3] R. Biswas, B. Limketkai, S. Sanner, and S. Thrun, "Towards object mapping in non-stationary environments with mobile robots," in *Intelligent Robots and System, 2002. IEEE/RSJ International Conference on*, vol. 1, 2002, pp. 1014–1019 vol.1.

[4] N. C. Mitsou and C. S. Tzafestas, "Temporal occupancy grid for mobile robot dynamic environment mapping," in *Control & Automation, 2007. MED '07. Mediterranean Conference on*, 2007, pp. 1–8.

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

[6] P. Biber and T. Duckett, "Dynamic maps for long-term operation of mobile service robots," *Proc. of Robotics: Science and Systems (RSS)*, 2005.

[7] ——, "Experimental analysis of sample-based maps for long-term SLAM," *The International Journal of Robotics Research*, vol. 28, no. 1, pp. 20–33, 2009.

[8] K. Konolige and J. Bowman, "Towards lifelong visual maps," in *IROS*, 2009, pp. 1156–1163.

[9] D. Meyer-Delius, "Probabilistic modeling of dynamic environments for mobile robots," Ph.D. dissertation, University of Freiburg, 2011.

[10] G. Wyeth and M. Milford, "Towards lifelong navigation and mapping in an office environment," *Proceedings of the 14th International Symposium of Robotics Research (ISRR)*, 2009.

[11] D. Fox, W. Burgard, S. Thrun, and A. B. Cremers, "Position estimation for mobile robots in dynamic environments," in *In Proc. of the National Conference on Artificial Intelligence (AAAI*, 1998.

[12] D. Hahnel, D. Schulz, and W. Burgard, "Map building with mobile robots in populated environments," in *Intelligent Robots and System, 2002. IEEE/RSJ International Conference on*, vol. 1, 2002, pp. 496–501 vol.1.

[13] C.-C. Wang, C. Thorpe, and S. Thrun, "Online simultaneous localization and mapping with detection and tracking of moving objects: theory and results from a ground vehicle in crowded urban areas," in *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, vol. 1, 2003, pp. 842–849 vol.1.

[14] E. Olson, "Real-time correlative scan matching," in *ICRA'09: Proceedings of the 2009 IEEE Intl. Conf. on Robotics and Automation*. Piscataway, NJ, USA: IEEE Press, 2009, pp. 1233–1239.

[15] J.-S. Gutmann and K. Konolige, "Incremental mapping of large cyclic environments," in *International Symposium on Computational Intelligence in Robotics and Automation*, 1999.

[16] 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, 2003, pp. 1899–1906 vol.2.

[17] X. Ji, H. Zhang, D. Hai, and Z. Zheng, "A decision-theoretic active loop closing approach to autonomous robot exploration and mapping," in *RoboCup 2008: Robot Soccer World Cup XII*, ser. Lecture Notes in Computer Science, L. Iocchi, H. Matsubara, A. Weitzenfeld, and C. Zhou, Eds. Springer, 2009, vol. 5399, pp. 507–518.

[18] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: incremental smoothing and mapping," *Robotics, IEEE Transactions on*, vol. 24, no. 6, pp. 1365–1378, 2008.

[19] H. Johannsson, M. Kaess, M. F. Fallon, and J. J. Leonard, "Temporally scalable visual SLAM using a reduced pose graph," Computer Science and Artificial Intelligence Laboratory, MIT, Tech. Rep. MIT-CSAIL-TR-2012-013, May 2012.