Topological Mapping for Manhattan-like Repetitive Environments

Sai Shubodh Puligilla*, Satyajit Tourani*, Tushar Vaidya*, Udit Singh Parihar*, Ravi Kiran Sarvadevabhatla and K. Madhava Krishna

Abstract—We showcase a topological mapping framework for a challenging indoor warehouse setting. At the most abstract level, the warehouse is represented as a Topological Graph where the nodes of the graph represent a particular warehouse topological construct (e.g. rackspace, corridor) and the edges denote the existence of a path between two neighbouring nodes or topologies. At the intermediate level, the map is represented as a Manhattan Graph where the nodes and edges are characterized by Manhattan properties and as a Pose Graph at the lower-most level of detail. The topological constructs are learned via a Deep Convolutional Network while the relational properties between topological instances are learnt via a Siamese-style Neural Network. In the paper, we show that maintaining abstractions such as Topological Graph and Manhattan Graph help in recovering an accurate Pose Graph starting from a highly erroneous and unoptimized Pose Graph. We show how this is achieved by embedding topological and Manhattan relations as well as Manhattan Graph aided loop closure relations as constraints in the backend Pose Graph optimization framework. The recovery of near ground-truth Pose Graph on real-world indoor warehouse scenes vindicate the efficacy of the proposed framework.

I. INTRODUCTION

This paper explores the role of topological understanding and the concomitant benefits of such an understanding to the SLAM framework. Figure-1 shows an erroneous Pose Graph (PG) labelled ‘a’, while the topological graph TG is shown labelled as ‘b’ in the same figure succinctly. Each node in the TG is labelled by the Deep Convolutional Network. The TG is converted to a Manhattan Graph (MG) wherein the Manhattan properties of the nodes (length or width of the topology) and edges are gleaned from the PG. While the MG facilitates seamless loop detection between a pair of Manhattan nodes, such relations when integrated with a backend SLAM framework, enable the recovery of an optimized pose-graph and corresponding map. The crux of the paper lies in detailing the framework and its efficacy in challenging real world settings of two different warehouses.

There have been a number of works in this area and a detailed review of such methods can be seen in [1]. Prominent and well cited amongst these include [2]–[7]. Most of these methods are focused exclusively on vision based loop detection with invariant descriptors. Many relate to an individual image as a distinct topology of the scene without relating such nodes to a meta-level label such as a rackspace, corridor, intersection etc. The classification task of place categorisation has been extensively performed on large datasets such as [8]–[10]. Works such as [11], [12] and more recently, [13], [14] present hierarchical scene representations and how they are useful for various navigation tasks. However, neither of these exclusively tackle highly challenging repetitive environments such as warehouses.

[15] shows the use of topological constructs for robot navigation in a simulated environment with the help of graph neural networks, where topological features are embedded into the nodes of a graph network. However, the method relies on the availability of noise-free sensor input, along with distinct topologies. [16] shows how a Bayesian inference over topologies can be performed to obtain more accurate topological maps. However, it does not entertain notions of meta-level topological labels that go beyond an immediate lower-level topology restricted to the scene seen by the robot.

In this paper, we distinguish ourselves by portraying how higher level/meta level topological constructs that go beyond an immediate frame/scene and the relations that they enjoy amongst them percolate to a lower level pose-graph and elevate their metric relations. In fact, we recover close to ground truth floor plans from a highly disorganized map at the start. This is the essential contribution of the paper. In addition, the following constitute our contributions:

1) A deep convolutional network capable of learning warehouse topologies.
2) A Siamese Neural Network based relational classifier which resolves topological element ambiguity and helps achieve an accurate pose graph purely based on Topological relations.
3) We showcase a backend SLAM framework that integrates loop closure relations from an intermediate level Manhattan Graph to the lowest level Pose Graph and elevate a disoriented unoptimized map to a structured optimized map which closely resembles the floor plan of the warehouse. Apart from the loop closure relations, the SLAM integrates other Manhattan relations to the pose graph. Ablation studies show the utility of both loop and Manhattan constraints as well as the superior performance of an incremental topological SLAM over a full batch topological SLAM. (Refer to Table IV.)
4) We also show how the two-way exchange between the MG and PG further improves the accuracy of the PG. This two-way exchange between the various levels of representation is unique to this effort. Refer to the bottom two rows in Table IV.

Through the above formulation, the paper essentially...
exploits the Manhattan properties present in indoor warehouse scenes to perform PG recoveries. Project page: https://github.com/Shubodh/ICRA2020.

II. METHODOLOGY

Consider an unoptimized pose graph $PG$ represented by its nodes as $V_{pg}$ and edges as $E_{pg}$. The edge relations are of the following kinds:

- Odometry relation between successive nodes.
- Loop closure relation between a pair of nodes.
- Manhattan relation between a pair of nodes.

We obtain odometry relations from fused ICP and wheel odometry estimates which gives us the initial pose graph, which is highly erroneous. We then leverage the topological and Manhattan level awareness to generate the loop closure and Manhattan relations and use them for pose graph optimization to recover accurate graphs. This whole process is divided into 3 sub-sections:

1) Topological categorization using a convolutional neural network classifier and its graph construction.
2) Constructing Manhattan Graph from the obtained Topological Graph and predicting loop closure constraints using Multi-Layer Perceptron.
3) Pose graph optimization using obtained Manhattan and loop closure constraints.

Each part of the pipeline is described in each sub-section below, followed by experiments and results which are explained in the next section.

A. Topological Categorization and Graph Construction

Every node $V_i \in V_{pg}$ is associated with a topological label $L(V_i)$ or $L_i$ where $L_i \in L = \{rackspace, corridor, transitions\}$ for a warehouse scene. To obtain these topological labels from visual data, we train a Convolutional Neural Network (CNN) configured for classification. The training data consists of RGB images resized to $224 \times 224$ and paired with topological node labels. For our warehouse setting, the labels are Rackspace, Corridor, Intersection:

- Rackspace: Location on path between two racks
- Corridor: Location on the warehouse boundary path common to racks
- Intersection: A transition location on the path

Figure-1 entails examples of frames and their topological labels. Using PyTorch framework [17], we train a ResNet-18 [18] architecture pre-trained on ImageNet [19] with its final layer replaced by a 3-neuron fully connected layer, corresponding to the possible topological node labels. During training, we optimize the network to minimize cross-entropy loss. To account for class imbalance, we use class-weighted loss [20] with the following set of weights: Rackspace: 2.48, Corridor: 2.16, Intersection: 7.38. The CNN is fine-tuned for 356 epochs using Adam optimizer with a learning rate of 0.001 for the pre-trained ResNet-18 layers and a learning rate of 0.005 for the final layer weights. We stop training when the validation loss starts to increase. For training, we use 33,273 images from two warehouses with a mini-batch size of 8. To evaluate the trained network, we use 21,200 images. The results are presented in the next section.

After obtaining the inferred labels from the CNN, we group together the adjacent nodes that share the same label. Thus, a node in Topological Graph $TG$ consists of two positions from the dense Pose Graph $PG$, i.e. the starting and ending positions of that topology.

B. Manhattan Graph Construction and Constraint Prediction using MLP

We now explain how the Topological Graph $TG$ of the last section is converted to a Manhattan Graph, $MG$. We denote each node in the $MG$ as a meta-node, $M_i \in MG$, where $M_i$ corresponds to a collection $C$ i.e. $V_i, V_{i+1}, ..., V_{i+n}$ of $PG$ nodes, such that we write $V_i \in C(M_j)$ and $L(M_j) = L(V_i)$.
angles \( \phi \)

e.g., \( L \)

are the same instance of a topological construct. In other words, the MLP determines if any two nodes in \( MG \) are the same instances only if the nodes correspond to the same topological instance. We base our approach on two strong assumptions:

- Each node comprises of one contiguous region of one particular category.
- Each node has displacement only in one direction. (Along x or y).

The classification that results from the MLP is particularly powerful due to its ability to classify two topological instances to be the same even when viewed from opposing viewpoints. This is shown in Figure-2 where the same topology is viewed from opposite viewpoint and have little in common. Yet the MLP’s accurate classification of them to be the same instance becomes particularly useful for the Pose Graph optimization described in the next section.

The MLP’s non-reliance on perceptual inputs also comes in handy for repetitive topologies. Warehouse scenes are often characterized by repetitive structure and are prone to perceptual aliasing. The classification accuracy of the MLP is unaffected by such repetitiveness in the environment since it bypasses perceptual inputs. Yet the MLP does make use of perceptual inputs minimally in that it attempts to answer if the two nodes in the MG are the same instances only if the topological labels of the two nodes are predicted to be the same by the CNN.

C. Pose Graph Optimization

The Manhattan relation that exists between two nodes \( M_i, M_j \) and represented as \( R(M_i, M_j) \) =< \( \Delta x_{ij}, \Delta y_{ij}, \Delta \theta_{ij} \) > serves as a Manhattan constraint between the nodes corresponding to \( M_i, M_j \) in the pose graph (in a manner consistent with the edge relations given in posegraph libraries such as G2O [22], GT-SAM [23]). \( \Delta \theta_{ij} \) is typically 0° or 180° depending on whether the topology is being revisited with the same or opposing orientation.

The output of the MLP classifier is also used to invoke loop closure constraints. A pair of nodes classified to be...
the same topological construct by MLP corresponds to two sets of pose-graph nodes in the unoptimized graph belonging to the same area. Multiple loop closure relations are thus obtained between the pose-graph nodes of these two sets. Apart from these, there exist immediate Manhattan relations between two adjacent rackspaces or two adjacent corridors or a rackspace adjacent to a corridor mediated through an intersection. All such relations that exist in the Manhattan Graph as well as the loop closure relations percolate to the nodes in the PG as described further below.

In effect the optimizer solves for [24]:

\[
X^* = \arg\max_X P(X|U) = \arg\max_X \prod_i P(x_{i+1}|x_i, u_i) \times \prod_{i \in C(M_i), j \in C(M_j)} P(x_j|x_i, c_{ij}) \times \prod_{i \in N(M_i), j \in N(M_j)} P(x_j|x_i, m_{ij})
\]

where \(P(X|U)\) is posterior probability of posegraph \(X\) over set of constraints \(U\), \(x_i\) and \(u_i\) are \(i\)th pose and controls of the robot. The loop closure relation \(c_{ij}\) between nodes \(i\) and \(j\) is obtained using ICP. There are in principle \(n(C(M_i)) \times n(C(M_j))\) loop closure relations that are possible between topological constructs \(M_i\) and \(M_j\) where \(C(M_i)\) is the collection set of Manhattan node \(M_i\) as described before and \(n(P)\) is the cardinality of the set \(P\). Whereas in practice we only sample a subset of such relations to constrain the graph.

Similarly, the graph is also constrained by Manhattan relations \(m_{ij}\) that are invoked between the pose-graph nodes that constitute the sets \(N(M_i)\) and \(N(M_j)\) where \(N(M_i)\) and \(N(M_j)\) represent the Pose Graph nodes within the neighbourhood of \(M_i\) and \(M_j\).

Typically \(N(M_j) \subseteq C(M_j)\). More formally, let \(S\) be the set that enumerates all loop closure pairs discovered by the MLP over a Manhattan Graph MG. i.e \(S = \{ (M_i, M_j), (M_i, M_k), \ldots, (M_p, M_q) \}\), where each element of the set is a loop closure pair on the graph and \(\{M_i, M_j, \ldots, M_q\}\) are the nodes of the MG. Let \(i\) be an iterator iterating over the element of \(S\), \(S(i) = (M_p, M_q)\). Let \(LC\) be the set of all loop closure relations, \(c_{ij}\) obtained for every \(S(i) \in S\) by sampling from the \(n(C(M_i)) \times n(C(M_j))\) number of loop closures possible for every \(S(i) \in S\). Similarly, let \(M\) be the set of all Manhattan relations \(m_{ij}\) obtained for every \(S(i) \in S\) from the neighbouring nodes in the unoptimized graph \(N(M_i), N(M_j)\) for every \(S(i) = (M_i, M_j) \in S\). Then

\[
X^* = \arg\max_X \prod_i P(x_{i+1}|x_i, u_i) \prod_{c_{ij} \in LC} P(x_j|x_i, c_{ij}) \prod_{m_{ij} \in M} P(x_j|x_i, m_{ij})
\]

There were two experiments performed. First, we sample our training data according to the layout and length constraints of warehouse-1 and use the data-points of warehouse-1 as the lone testing data. In our second experiment, we train
our MLP specifically according to the layout and length constraints of warehouse-2.

The detection of nodes belonging to the same topology was observed to be accurate at the initial phase of the trajectory. The latter part of the trajectory was not accurate and had drift due to which the detection of node-pairs was observed to be inaccurate (Unoptimized Pose Graph shown in Figure-3). We were able to improve the accuracy of the MLP and were able to generate accurate loop pairs by performing optimization on the Pose Graph in a cyclic fashion as shown in Figure-3.

We performed the experiments on both warehouses. The accuracy is calculated by checking for the percentage of the true node pairs. An accuracy of 71.2% and 67.7% was observed for the first and the second warehouse respectively.

C. Pose graph optimization Results

The ablation study on the type of constraints have been done in five stages. The robustness of map recovery increases with each stage which reflects in Absolute Trajectory Error in Table IV. The experiments are conducted on four trajectories with varying lengths and starting deformation. Trajectories $W_1$ and $W_2$ are from the first warehouse and $W_3$ and $W_4$ are from the second warehouse in Table IV. Qualitatively, the initial trajectories are shown in first row of Figure 5.

1) Stages of Map Recovery:

(i) Manhattan constraints: Only manhattan constraints are used to optimize pose graph, PG. Constraints are extracted from manhattan graph, MG, between nodes proposed by MLP to be similar.

(ii) Loop Closure and Manhattan constraints: Apart from Manhattan constraints, Loop Closure constraints as explained in section II-C are also used to constrain the PG.

(iii) Dense Proposals from MLP: We consider nodes that have been classified to belong to the same instance with low confidence along with those classified to be the same with high confidence. This increases the number of constraints improving the optimization performance. The wrongly detected loops are filtered based on the loop closure (ICP) residual cost and do not make it to the optimization.

(iv) Dense Proposals by MLP in Feedback Loop: A feedback loop is invoked on the optimized PG from previous stage. A new MG is computed on the optimized PG, this manhattan graph, MG, is fed to MLP and sets of constraints are generated in a cyclic manner. This feedback mechanism leads to MLP performance improvement as shown in Table V and also helps in achieving very low Absolute Trajectory Error, ATE of 1.82 meters on four different maps from 11.57 meters in unoptimized map. This corresponds to the fourth of the contribution mentioned in the Section I.

(v) Incremental formulation: Performing the feedback strategy from the previous stage in an incremental formulation in ISAM [23] helps us to achieve the lowest ATE of 1.45 meters in our system. This confirms the robustness of our system to recover from highly unoptimized trajectories.

2) Qualitative Results: We evaluate our system in two challenging real warehouse settings. The warehouse dimensions are $30m \times 50m$ and contains 21 rack-spaces with intermediate corridors and intersections. All experiments start with highly deformed trajectories. In all the cases, we were able to recover trajectories close to the groundtruth. Note that in our case, the ground-truth trajectory is the optimized map from the cartographer that has been confirmed with warehouse floor plan by our collaborators. These results are shown in Figure 5. The top row shows highly distorted pose graph trajectories while the middle row showcases the results of our optimization framework. The last row depicts the ground truth trajectories. The overall pipeline gets best illustrated with Figure 3.

D. Improving the performance of the state-of-the-art SLAM system: RTABMAP

We compare our topological SLAM pipeline with the state-of-the-art SLAM system RTABMAP [25], which is a highly modular library with the integration of various sensors like monocular camera, stereo camera, LiDAR, IMU and wheel odometry. When we evaluated RTABMAP on our warehouse dataset, we found that RTABMAP detects many False Positive loop closure constraints owing to similar looking corridors, and thus, incorrectly merges different parallel corridors into the same corridor. By incorporating our topological constraints in RTABMAP, we achieved better trajectory in terms of Absolute Trajectory Error, as shown in Table III. This increase in performance is attributed to the non-reliance of MLP on individual frame-wise visual input but instead, it is utilizing the geometric structure of the topological representation. Hence, our topological constraints can be used to extend traditional Visual SLAM pipeline in highly repetitive Manhattan-like environments. Further qualitative results can be found in the Project page.

| TABLE III | ATE COMPARISON OF TOPOLOGICAL SLAM WITH RTABMAP |
|---------------------------------|---------------------------------|
| RTABMAP                        | RTABMAP + Topological Constraints |
| 4.45                           | 3.36                           |

E. Robustness Analysis

We analyze the performance of the topological SLAM due to errors in topological classification due to the CNN and due to failure to detect loops by the MLP. Errors in
topology classification manifest as loop detection in the MG. Therefore, the analysis is one of the robustness due to wrong loop detection wherein both false positive and false negative cases are considered. The robustness to the pose graph optimization stems due from the following features:

1) Residuals in the ICP estimated loop closure end up serving as priors to the element of dynamically scaled covariance matrix [4], which serves as a robust kernel providing for backend topology recovery even when the number of wrong loop closures increase.

2) An optimized PG feeds back to the MG and alleviates its error. The improved MG improve the loop detection performance of MLP, which percolate to the PG nodes and further improve its accuracy. Overtime this iterative exchange of information between the various representations improves the robustness of the PG backend.

To analyse the performance of our robust kernel exclusively in presence of the outliers, we synthetically introduced false positive (FP) and false negative (FN) loop closure pairs in the constraints for $PG$. In Figure 4 X-axis represents percentage of loop closure pairs in the data-set having equal amount of FP and FN pairs. Y-axis represents ATE of TG with respect to the ground truth. The Figure 4 depicts the performance of our framework due to errors occurred in topology classification and loop detection.

From the analysis of results in Figure 4 it is evident that gradual increase in outliers can be tolerated by Robust kernel with DCS [4] as compared with non robust optimization techniques.

### Table IV

| Method Type | W-2.1 ATE | W-2.2 ATE | W-1.1 ATE | W-1.2 ATE | Avg ATE |
|-------------|-----------|-----------|-----------|-----------|---------|
| Unoptimized | 4.7       | 7.5       | 16.3      | 17.8      | 11.57   |
| MLP Manhattan (G2O) | 3.42 | 2.85 | 4.5 | 7.4 | 4.54 |
| MLP Manhattan + LC (G2O) | 3.09 | 2.7 | 3.9 | 1.67 | 2.84 |
| Dense MLP Manhattan + LC (G2O) | 1.98 | 1.96 | 2.75 | 1.65 | 2.08 |
| Dense MLP Manhattan + LC (In Feedback Loop) (G2O) | 1.67 | 1.8 | 2.21 | 1.6 | 1.82 |
| Dense MLP Manhattan + LC (In Feedback Loop) (iSAM) | 1.6 | 0.98 | 1.02 | 2.2 | 1.45 |

### Table V

| Method Type                  | True Positive | False Positive | Accuracy |
|------------------------------|---------------|----------------|----------|
| MLP                          | 119           | 81             | 59.5     |
| MLP in Feedback Loop         | 133           | 55             | 70.7     |

### IV. CONCLUSION

This paper shows how higher level abstractions of an indoor workspace such as real warehouses can be used to effectively improve lower level backend modules of localization and mapping. Specifically we show how higher and intermediate level abstractions in the form of Topological Graph and Manhattan Graph can recover from backend pose graph optimization failures. Further by constant information exchange between the various levels of map abstractions we improve quantitatively the ATE by more than 87.4% starting from very distorted pose graphs. We further show the method is robust to failures in the higher level representations such, which occurs when the Deep CNN architecture wrongly classifies a topological construct or when the Siamese style classifier wrongly detects or fails to detect loops in the Manhattan graph. The results shown are on two different real warehouse scenes over an area of around $30m \times 50m$, filled with many repetitive topologies in the form of corridor areas and rackspaces. Future results are intended to be shown on a variety of indoor topologies and office spaces such as for example those found in the Gibson environment [27].
