BoxMap: Efficient Structural Mapping and Navigation

Zili Wang, Christopher Allum, Sean B. Andersson, and Roberto Tron

Abstract— **While humans can successfully navigate using abstractions, ignoring details that are irrelevant to the task at hand, most of the existing approaches in robotics require detailed environment representations which consume a significant amount of sensing, computing, and storage; these issues become particularly important in resource-constrained settings with limited power budgets. Deep learning methods can learn from prior experience to abstract knowledge from novel environments, and use it to more efficiently execute tasks such as frontier exploration, object search, or scene understanding. We propose BoxMap, a Detection-Transformer-based architecture that takes advantage of the structure of the sensed partial environment to update a topological graph of the environment as a set of semantic entities (rooms and doors) and their relations (connectivity). The predictions from low-level measurements can be leveraged to achieve high-level goals with lower computational costs than methods based on detailed representations. As an example application, we consider a robot equipped with a 2-D laser scanner tasked with exploring a residential building. Our BoxMap representation scales quadratically with the number of rooms (with a small constant), resulting in significant savings over a full geometric map. Moreover, our high-level topological representation results in 30.9% shorter trajectories in the exploration task with respect to a standard method. Code is available at: bit.ly/3F6w2Yl.**

I. INTRODUCTION

Navigating complex and unknown environments to achieve various tasks is a remarkable ability shared by both humans and animals. This capability is derived in part from the ability to use past experiences to predict salient environmental features and identify the most effective strategies for success [1]–[3]. Developing a similar ability for robots would be a valuable skill in real-world settings.

Current hierarchical abstractions that use semantic entities and physical relations have demonstrated success in long-term mapping, localization [4]–[6], and planning [7], [8]; however, they typically require extensive data collection via high-performance, resource-intensive sensors, alongside computationally demanding online algorithms. These requirements make such methods impractical for scenarios with limited resources, such as when using small-scale robots like the Harvard Ambulatory MicroRobot (HAMR) [9] or during long-duration missions [10].

Traditional approaches to generate a topological map of the environment are typically based either on pixel-wise semantic segmentation [11]–[13] with the identification of physical relations between semantic entities [14], or rely on the identification and clustering of wall entities [5], [15]. These methods do not take advantage of prior information about the environmental structure and the accuracy of the map relies on the post-processing procedures. Our previous work [16] proposed a multi-task learning architecture for multi-level abstractions; however, the topology generation was still based on ad-hoc post-processing of a low-level representation (occupancy maps).

Motivated by these issues, in this paper we propose BoxMap, an end-to-end method that transfers low-level raw measurements to topological maps (i.e., high-level semantic representations) of a structured environment by leveraging the predictability of its layout, such as indoor spaces comprised of polygonal rooms and with connecting doors.

With the development of deep learning, the prediction of polygonal instances typically involves two sequential steps *1)* predicting bounding boxes containing each polygon based on anchors or proposals, and *2)* detecting polygon corners inside a representative bounding box, by using either a Convolutional Neural Network (CNN)-Recurrent Neural Network (RNN) [17], or CNN-Graph Convolutional Network (GCN) paradigm [18], [19]. The resulting methods rely on many hand-crafted components, e.g., Non-Maximum Suppression (NMS) or anchor generation.

To alleviate the reliance on hand-crafted components, we can use DEtection TRansformer (DETR), an end-to-end method that directly predicts the set of bounding boxes, with a CNN backbone (to extract image features), a transformer encoder-decoder architecture (to eliminate anchor generation) and a set-based global loss that forces unique predictions via bipartite matching (to eliminate NMS) [20]. Later work built upon a DETR to extract non-rectangular polygons by adding a polygon regression head, a corner classification head [21], a multi-resolution representation, and extra post-processing. Unlike these prior works that use bounding boxes to *detect* objects, BoxMap uses boxes as *primitives* (i.e. building blocks) to represent an environment.

As a concrete example for the approach we develop, we focus on an exploration task in an unknown environment. There are multiple approaches in the literature to achieve this particular task. Traditional methods typically build a geometric map from low-level representations (points, lines) using Simultaneous Localization and Mapping (SLAM), combined with a heuristic exploration method like frontier-based exploration [22]. In more recent work, environmental priors are used to predict unknown regions based on assumed structure (e.g., lines) [23] or with deep learning models [16], [24], or directly predict navigation cues [25]–[27]. The

Z. Wang is with the Division of Systems Engineering, C. Allum with the Department of Mechanical Engineering, and S.B. Andersson and R. Tron are with the Division of Systems Engineering and the Department of Mechanical Engineering, Boston University, Boston, MA 02215, USA. {zw2445,ctallum,sanderss,tron}@bu.edu.

This work was supported in part by NSF FRR-2212051 and the Center for Information and Systems Engineering at Boston University.

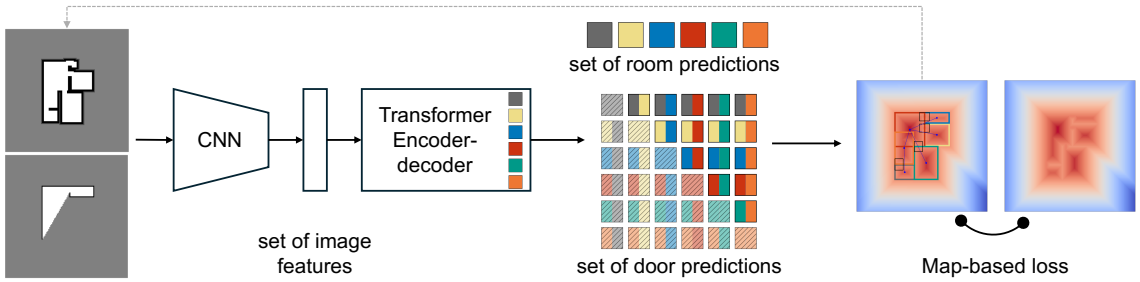


Fig. 1: Architecture diagram: the TSDF from the previous prediction is converted to an occupancy grid, which is concatenated with the new laser measurement to be fed into a DETR-based model (a combination of CNN and transformer) to give multiple embeddings. Each embedding is used to predict a room box and each pair of embeddings is used to predict a door box. These boxes are transformed into a single TSDF, which is then compared with the ground truth to update the model.

predictions are then used to enhance exploration efficiency.

Paper contributions. BoxMap models environments (rooms and doors) as boxes, together with their adjacency relations, and uses a DETR-like framework (Fig. 1) to achieve end-to-end topological graph prediction from low-level measurements. Specifically:

- We create interpretable embeddings (box representations) for the prediction of rooms and their physical relations, via a CNN feature extractor followed by an encoder-decoder transformer that models room arrangements.
- We use losses based on the Truncated Signed Distance Function (TSDF) of predicted boxes to learn against the ground truth TSDF. Compared to losses based on vertices, our method is easier to train and avoids the necessity of manual labelling.
- We introduce a hierarchical loss that improves the detection of small but topologically important details (such as doors); the loss subtracts large objects (rooms) from a TSDF to highlight and focus on small ones.

The application to exploration shows that our high-level graph representation enables:

- Semantic-level mapping by updating the graph with low-level measurements, yielding a significantly more compact memory footprint than a detailed map representation.
- Higher-level reasoning of the environment that leads to structured inferences even beyond observed regions. The resulting graph-based decision making leads to shorter paths than standard frontier exploration methods.

II. System Overview

We consider the problem of using a robot to explore an unknown environment from a random initial position with minimal map storage and total path traveled. We assume that: *1)* the robot is equipped with a laser range scanner; *2)* the robot can detect the map alignment (i.e., the predominant direction of walls using, e.g., the methods in [28], [29]); *3)* the robot has relatively accurate odometry.

The core idea of BoxMap is to leverage prior experience (a training set), and represent environments as graphs of boxes. The rest of this paper presents how our BoxMap representation is generated, and how it can be used for exploration.

Graph Prediction and Mapping. BoxMap represents rectangular rooms as boxes with four corners, non-rectangular rooms as overlapping boxes (which we refer to as multi-boxes), and doorways as square boxes connecting rooms. These are organized in a graph that is incrementally built from individual laser measurements via a learned model.

At time step t, from the robot position p_t, a laser measurement is acquired and converted to a local occupancy map, M_t^{laser}. This map is then concatenated with an occupancy map, $\hat{M}_{t-1}^{topo}$, instantiated from the topological graph of the last step, $\hat{\mathcal{G}}_{t-1}^{topo}$, to predict a new graph $\hat{\mathcal{G}}_t^{topo}$. This step uses TSDF as a tool to bridge the gap between topological and occupancy map information, effectively combining them. Overall, the mapping update process $f_{map}(\cdot)$ from BoxMap is represented as

$$\hat{\mathcal{G}}_t^{topo} = (\mathcal{V}, \mathcal{E}) = f_{map}(p_{t-1}, p_t, \hat{\mathcal{G}}_{t-1}^{topo}, M_t^{laser}), \quad (1)$$

where $\mathcal{V}$ is the set of nodes, $\mathcal{E}$ is the set of edges showing the spatial connections between nodes.

Graph-based exploration. We use the graph $\hat{\mathcal{G}}_t^{topo}$ to construct a planning graph $\hat{\mathcal{G}}_t^{nav}$, and then determine a frontier point g_t to steer to. Here, we define a candidate frontier on $\hat{\mathcal{G}}_t^{topo}$ as a room that has not yet been visited. We use the A* algorithm to find a feasible trajectory from p_t to g_t on an occupancy grid map M_t^{occ} that accumulates measurements up to the current time. Note that in our work, we keep our focus on the graph prediction, mapping and frontier selection, maintaining M_t^{occ} only in the A* planning for simplicity. However, it is not necessary, and in practice a local approach that uses only recent information should be sufficient. The graph is updated with a new measurement once the selected frontier has been reached, iterating the steps until a termination criterion is met.

III. Graph Prediction and Mapping

A. BoxMap Network Architecture

We use a DETR-based architecture as illustrated in Fig. 1 to implicitly represent prior structured information in the environment. It contains a CNN backbone, an encoder-decoder transformer, and task-specific prediction heads.

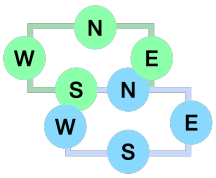


Fig. 2: The connectivity between two room boxes is validated by comparing the edges N, S, W, E of one room (in green) to the edges S, N, E, W of the other room (in blue). An entry will appear in the room box adjacency matrix if corresponding edges overlap, in this case green S and blue N.

1) Backbone: A CNN backbone is first used to extract lower-resolution image feature maps. It takes the concatenated local maps $\hat{M}_t^{topo}$ and M_t^{laser} and learns a common latent space representation. Each layer includes a downsampling convolution, followed by a standard convolution, both with Batch Normalization and Leaky ReLU. We use four layers with channel sizes 32, 64, 128, and 256.

2) Transformer: We use a standard transformer architecture to transform the feature maps into a set of embeddings. The transformer encoder processes the flattened image features and learned positional encodings as input and consists of four encoder layers. The transformer decoder is similarly composed of four decoder layers. Each encoder and decoder layer includes a multi-head attention mechanism and a feed-forward layer. The first decoder layer applies cross-attention, while the remaining layers use self-attention. The decoder produces M embeddings for M boxes in parallel, with box queries randomly initialized at the decoder's input.

3) Prediction heads: We define a box by its top left and bottom right coordinates $b^i = ((x_0, y_0), (x_1, y_1))$, together with an existence probability q^i. Embeddings from the transformer output are adopted to generate the prediction heads for rooms and doors. Each room head predicts: *1)* a probability specifying the box existence, and *2)* the room box coordinates. From the predicted room box coordinates, we create an adjacency matrix between boxes (see Fig. 2). Two additional door prediction heads generate door predictions from all possible outer sums of pairs of embeddings. Each door head predicts: *1)* a class specifying the door existence, masked by the adjacency matrix mentioned above, and *2)* the door box coordinates relative to the corresponding room edge.

B. Parametric TSDFs for Structured Shapes

Although there are multiple ways to represent a non-rectangular room as a multi-box, the TSDF of the shape is unique. Therefore, we propose to define an analytic map (based on Rectified Linear Units, $ReLU(x) = \max(0, x)$) from box coordinates b^i to a TSDF, and then base the training loss on TSDFs (Section III-C). In the following, $p = (x, y)$ represents a point in the environment, and γ is a fixed constant that defines the truncation value for a TSDF.

Box Representation. The TSDF representation of a $2D$ rectangle in the L_∞ norm is represented as the distance to the nearest wall along either the x or y axis,

$$f_{2D}(p) = \min(f_{1D,x}, f_{1D,y}); \tag{2}$$

the distance along one axis can be represented by multiple $ReLU$ functions:

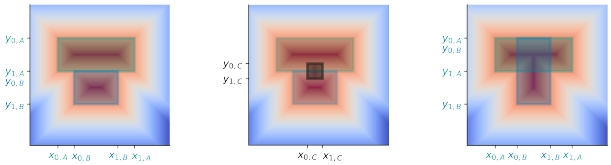
$$f_{1D,x}(x_0, x_1) = \min(ReLU(x - x_0 + \gamma) - ReLU(x - x_0 - \gamma) - \gamma, -ReLU(x - x_1 + \gamma) + ReLU(x - x_1 - \gamma) + \gamma); \tag{3}$$

$f_{1D,y}$ is obtained by substituting y for x in (3).

Room (Multi-Box) Representation. We represent a non-rectangular room as the merging of multiple overlapping boxes, and a doorway as the merging of a square-shaped box on the overlapping edges of two room boxes (Fig. 3). We can merge multiple SDFs using the $\max$ operation:

$$f_{composite}(p) = \max_i \left(q^i \cdot (f_{2D}^i(p) + \gamma) - \gamma \right), \tag{4}$$

where i is the index of a box.



(a) Room merge (b) Door merge (c) Non-rectangular

Fig. 3: Shape Merging Operation.

C. Loss Functions

We use two types of loss function to facilitate training: one based on TSDFs, the other on the box representation. Notably, our room and door predictions can be merged to form a TSDF prediction $\hat{y}_{tsdf}$ following (4), which can be compared to the ground truth y_{tsdf} from the dataset.

Map-Based Loss. We define a series of losses that compare TSDF representations. Thanks to our parametric TSDF representation (Section III-B), our approach can use raw occupancy maps for the training dataset (without the need for manual labeling of individual rooms), while still working directly on the high-level box parameters.

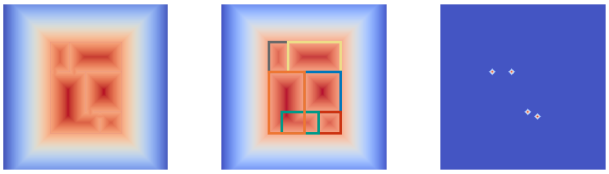
We start with a basic L_2 loss over pixels p:

$$l_{tsdf}(y_{tsdf}, \hat{y}_{tsdf}) = \|y_{tsdf} - \hat{y}_{tsdf}\|_2. \tag{5}$$

We also introduce an auxiliary boundary loss:

$$l_{tsdf,W}(y_{tsdf}, \hat{y}_{tsdf}) = \|y_{tsdf} - \hat{y}_{tsdf}\|_2^W, \tag{6}$$

to emphasize accurate wall predictions. The operator $\|\cdot\|_2^W$ indicates an L_2 norm with a mask following the ground truth wall boundaries (denoted W, where y_{tsdf} is zero). Empirically, the losses above do not capture doors well (due to their small size), in the sense that errors in the prediction of doors produce changes that are much smaller than errors in the prediction of rooms. We observe that the difference between the ground truth TSDF and the room-boxes-only TSDF, noted



(a) (b) (c)

Fig. 4: (a) Ground truth TSDF with boundary overlaid in black, (b) Predicted room TSDF, (c) Difference between (a) and (b) highlights the door locations.

as $\tilde{y}_{tsdf}^{doors}$, highlights the locations of the doors (Fig. 4). We therefore define a hierachical door loss $l_{\diamond doors}$ that, assuming an accurate prediction of the rooms, focuses on doors:

$$\begin{aligned} l_{\diamond doors}(\tilde{y}_{tsdf}^{doors}, \hat{y}_{\diamond}^{doors}) &= \|\tilde{y}_{tsdf}^{doors} - \hat{y}_{\diamond}^{doors}\|_2^{\tilde{W}}, \\ \hat{y}_{\diamond}^{doors}(p) &= \max_i \ ReLU(s^i/2 - ||p - c^i||_1), \end{aligned} \tag{7}$$

where $\hat{y}_{\diamond}^{\text{doors}}$ is the TSDF obtained by maximizing over individual boxes, with c^i and s^i being the centroid and dimension of a door box. The operator $\|\cdot\|_2^{\tilde{W}}$ indicates an L_2 norm with heavier penalties near wall boundaries ($\hat{y}_{tsdf}$ is near zero) and the wall of the input (denoted $\tilde{W}$).

The total map-based loss $\mathcal{L}_{TSDF}$ is defined as the mean, over all pixels p, of the losses in (5), (6), (7).

Box-Based Loss. We include a self-supervised IOU loss to measure overlap between room boxes (denoted as $\mathcal{V}$):

$$\mathcal{L}_{IoU}(\mathcal{V}) = \frac{1}{M(M-1)} \sum_{A \in \mathcal{V}} \sum_{B \in \mathcal{V}/A} IoU(A, B). \tag{8}$$

The gate loss on the set of existence probability of boxes $\mathcal{Q}$ is proposed as

$$\mathcal{L}_{Gate}(\mathcal{Q}) = \frac{1}{|\mathcal{Q}|}(\sum_{\mathcal{Q}} q^i(1-q^i) + \sum_{\mathcal{Q}} q^i), \tag{9}$$

where the first term is a bi-modal regularizer to force q^i to be either 0 or 1, and the second term emphasizes sparsity.

Final Loss. The final loss is

$$\mathcal{L}_{total}(\hat{y}, y, \mathcal{V}, \mathcal{Q}) = \mathcal{L}_{TSDF}(\hat{y}, y) + \mathcal{L}_{IoU}(\mathcal{V}) + \mathcal{L}_{Gate}(\mathcal{Q}). \tag{10}$$

D. Topological Graph Generation

A topological graph $\hat{\mathcal{G}}_t^{topo}$ is generated from the model outputs as follows: *1)* The rooms $\mathcal{V}$ in the environment are identified from the predicted room boxes, where a rectangular room is represented as a single node, and a non-rectangular room is represented as multiple overlapping box nodes with edges added between each pair without door information. Each node contains the location and dimension of the box. A pair of room boxes are connected with an edge if there exists a door connecting them. *2)* The doors are identified from the predicted door boxes. Each door contains the location, dimension and the connected room pair of the box. An edge is added to the graph if the points on opposite sides of the candidate door box are traversable within the door region, based on the accumulation of $\hat{M}_{t-1}^{topo}$ and M_t^{laser}.

E. Dataset

We used environments from the RPLAN dataset [30], which contains 80k residential building floorplans. The TSDF for each environment was generated using the chamfer distance transform [31]. We selected 400 environments (each with five rooms) for training, and 41 for testing. Note that the choice of five rooms was made to keep the setting manageable, allowing for easy interpretation of results. Preliminary results (not shown for space reasons) indicate that our architecture can handle more complex environments with more rooms.

The robot was driven from random positions to implement standard frontier exploration in unknown environments to create the dataset. Each input and ground truth TSDF was obtained by cropping a 128×128 local accumulated occupancy map. The model was initially trained on the occupancy maps, followed by fine-tuning with a dataset consisting predicted environments concatenated to the measurements.

IV. Graph-based Exploration

In this section, we describe our algorithms for semantic exploration (Fig. 5) on graphs.

A. Planning Graph Generation

While the topological graph $\hat{\mathcal{G}}_t^{topo}$ (Fig. 6a) is a compact representation of the environment, we post-process it as $\hat{\mathcal{G}}_t^{nav}$ (Fig. 6b) to aid path planning. Specifically, we use the following steps: *1)* Add a node for each door, connecting it with edges to the corresponding room pair. *2)* Connect separate doors in the same room with an edge. *3)* Assign a position attribute for each rectangular room node at a point interpolated between the room centroid and its associated door centroid. We implement the interpolation by assuming that by entering a close-to-door region in a small room, the robot can capture the key features of the room (including itself and the interconnecting area) and make a reliable prediction. The planning graph is further extended to dynamically connect the robot node with the room-box node it resides in, as well as with the neighboring box nodes (Fig. 6c).

B. Semantic Exploration

We propose two different exploration strategies, one greedy and one that considers long-term performance using the global environment structure hallucinated from our model. We first identify previously visited nodes where measurements were taken. Both algorithms iteratively update $\hat{\mathcal{G}}_t^{nav}$ with new measurements once a selected frontier has been reached, and terminate when there are no more unvisited rooms.

(C1) **Greedy strategy:** The robot uses the closest unvisited room on $\hat{\mathcal{G}}_t^{nav}$ as the interim goal g_t .

(C2) **Receding Horizon (RH) strategy:** At each iteration, the robot first plans the shortest path on $\hat{\mathcal{G}}_t^{nav}$ which visits all unvisited rooms at least once using Dynamic Programming. Then the robot selects the first room node on the generated path as the interim goal g_t.

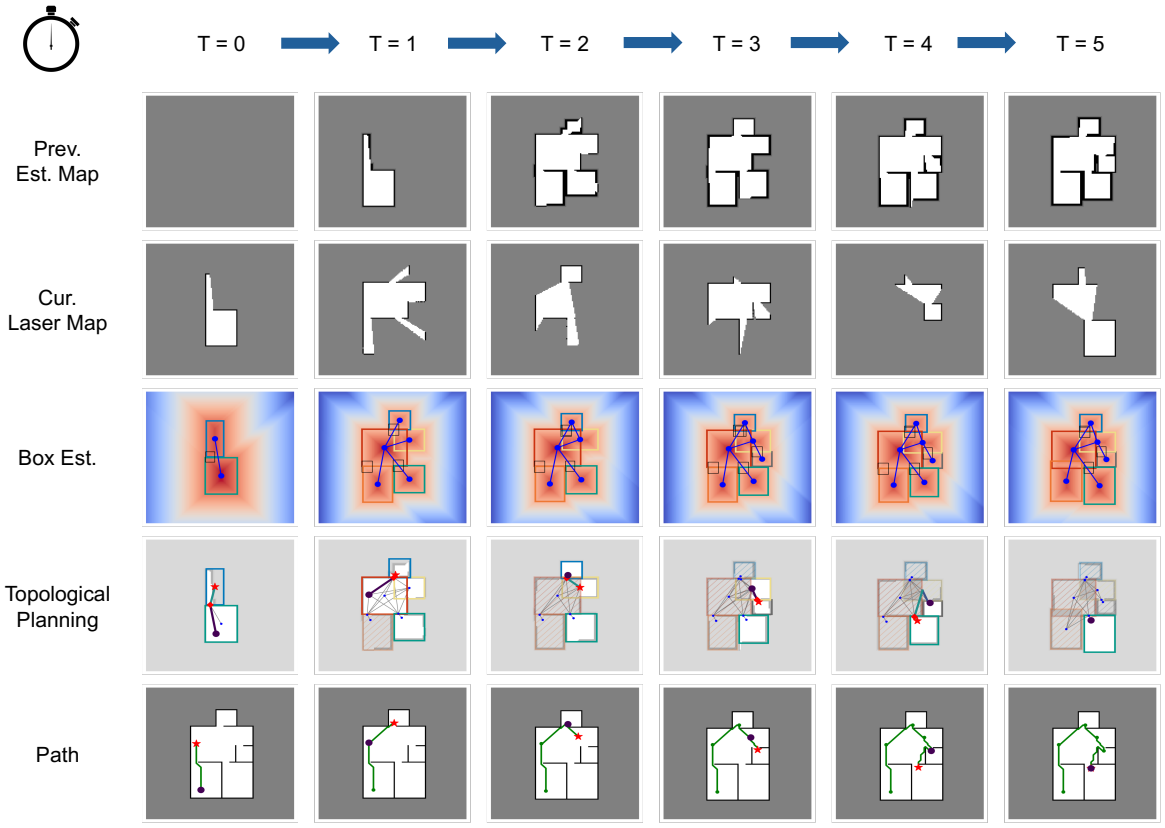
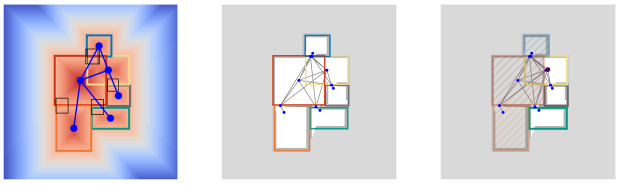


Fig. 5: The progression of the agent exploration over time. It starts by instantiating the last estimated graph (row 1), then collects the laser measurement (row 2), and centers them to the network. The estimated room and door boxes (row 3) are used to construct a pose graph and plan the next move (row 4, shading indicates the room has been visited, overlaid with the predicted map). On row 4 and 5, the purple dot is the robot pose, red star is the point-to-go and red diamond is the nearest door. On the ground truth floorplan (row 5), green dots are the pose history of laser measurements.



(a) Topological $\hat{\mathcal{G}}_t^{topo}$ (b) Planning $\hat{\mathcal{G}}_t^{nav}$ (c) Extended $\hat{\mathcal{G}}_t^{nav}$

Fig. 6: Planning graph generation.

V. Simulations

A. Simulation Setup

The simulations for the training dataset and algorithm evaluations use PseudoSLAM [32]. The laser scanner has a 360° field of view and 9 m range. All environments were represented as 2D occupancy maps with a resolution of 0.14 m per pixel.

B. Baselines

We compare against two geometric-map based frontier exploration methods, where a frontier pixel is defined as the centroid of the segment that separates known regions from unknown regions. To give a fair comparison with our proposed method, the geometric map only accumulates laser measurement when a selected frontier has been reached.

(C3) **Traditional Frontier Exploration:** Candidate frontier pixels are selected using the method from [33]. Specifically, we define a reward $R(p_f)$ that combines path cost and information gain for each frontier candidate pixel p_f:

$$R(p_f) = \lambda I(p_f) - \|p_f - p_t\|_2, \tag{11}$$

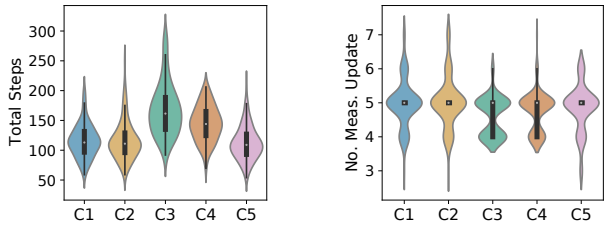
where $\|\cdot\|_2$ indicates the standard Euclidean metric, $I(p_f)$ is the information of the frontier pixel defined as the number of unknown pixels within the range of the sensor around the pixel, and λ is a tuning weight. The frontier candidate with the highest reward is selected as the next goal point. The process is repeated until there are no more frontier pixels.

(C4) **Frontier Exploration with Map Completion:** We also consider a data-driven baseline [34] where the candidate frontier pixels are selected based on the completed geometric map. We complete the partial map using the model trained in [16]. We find all the connected pixels that are unknown in the current map M_t^{occ} but are free according to the completed map $\hat{M}_t^{occ}$; the number of such pixels is used as $I(p_f)$.

(C5) **Hybrid Strategy:** For completeness, we add a strategy using the topological graph for exploration, while using the

TABLE I: Evaluation results. ↓ indicates that smaller values imply better performance. $N = 256$ and $M = 6$ here.

	(R1) ↓	(R2) ↓	(M1) ↓	(M2) ↑	(M3) ↓
BoxMap Greedy (C1)	114.9	5	$\mathcal{O}(M^2)$	0.96	2.5%
BoxMap RH (C2)	115.2	5	$\mathcal{O}(M^2)$	0.96	2.4%
Occ. Standard (C3)	166.4	4.6	$\mathcal{O}(N^2)$	1	0
Occ. NN (C4)	143.6	4.8	$\mathcal{O}(N^2)$	1	0
Occ. RH (C5)	111	4.9	$\mathcal{O}(N^2)$	0.98	1.3%



(a) Total steps (b) Measurement updates

Fig. 7: Violin plot over all runs of (a) total steps and (b) total measurement updates. Violin plots illustrate data distributions by superimposing kernel density plots onto box plots.

entire geometric map M_t^{occ} for graph predictions.

C. Evaluation of Semantic Exploration Performance

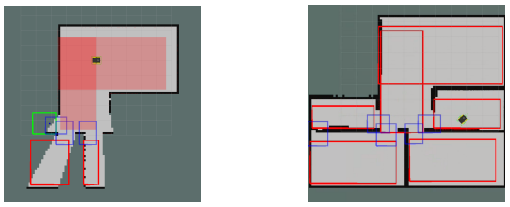
We ran the algorithms on 41 test environments, from three random initial positions each. To evaluate performance we define the following metrics averaged over all runs.

(R1) Total steps: number of pixels traversed to complete.
(R2) Number of measurement updates.

We consider three additional metrics that are not directly related to the exploration but are useful to assess the differences between our algorithms and the baselines.

(M1) Map memory.
(M2) Structural Similarity Index Measurement (SSIM) between the final TSDF and the ground truth map.
(M3) Hamming loss: pixel-wise classification accuracy of the final map.

Our results, summarized in Table I and Fig. 7, show that our method is superior to the baseline, yielding a more compact map representation as well as shorter trajectories. This indicates that our method can leverage semantic prior knowledge learned from data to create a higher-level representation from low-level partial measurements, which result in improved performance; such information can not be directly obtained from any geometric-based frontier methods. Moreover, despite the reduction in storage, the maps that can be reconstructed by our methods show only a minor reduction in accuracy with respect to geometric maps (as indicated by metrics (M2) and (M3)). It should be noted that our method discards low-level geometric information other than the current measurement. This is achieved by converting $\hat{\mathcal{G}}_{t-1}^{topo}$ to $\hat{\mathcal{M}}_{t-1}^{topo}$, and using $\hat{\mathcal{M}}_{t-1}^{topo}$ as the new model input. While there are potential prediction and conversion errors, the model may miss some visited semantic entities in the long horizon. We have observed, however, that the model is able to correct the error if the affected area is revisited (at the cost of longer trajectories).



(a) Init. Pred. (b) Final Pred.

Fig. 8: ROS Simulation: (a) The initial LiDAR scan of the Jackal with room (red and green rectangles, green indicated the selected frontier) and door (blue rectangle) predictions, shaded-in red indicates the room that robot starts in. (b) Final predictions after having explored the entire map.

D. Simulation in Gazebo

We further tested our exploration algorithm using the Robot Operating System (ROS) in the Gazebo simulator with a Jackal robot (Clearpath Robotics) equipped with a LiDAR sensor, leveraging the `gmapping` package to accumulate the laser measurements. Accounting for the robot's footprint, to avoid collisions with obstacles, we add waypoints near doors to traverse smoothly these narrow regions. Fig. 8a demonstrates an initial random placement of the robot in an apartment layout, the initially constructed graph and the selected frontier. The algorithm is repeated until all the rooms have been visited. Fig. 8b shows the graph $\hat{\mathcal{G}}^{topo}$ when the algorithm terminates. The result shows that BoxMap is robust to imperfect box predictions due to noisy sensor and odometry measurements. Simulations in multiple environment layouts (not shown due to space constraints) confirmed these findings.

VI. CONCLUSIONS

We propose a CNN-transformer-based architecture to learn and update topological information of the environment from low-level measurements, which significantly reduces the map storage in navigational execution. To facilitate the training process, we propose to learn from a TSDF-map instead of box vertices and demonstrated its strength on learning semantic entities and relations. Through simulations we demonstrate that our graph-based semantic exploration algorithm achieved better performance compared to geometric-map-based frontier exploration algorithms.While our algorithms assumed both perfect odometry and sensing, our simulations in Gazebo indicated that it can be adapted to realistic settings.

Future work includes improving the robustness of the architecture by either training with cluttered environments, combining with obstacle removal algorithms, or exploring other DETR models (e.g. Deformable DETR). Further testing would involve real hardware, evaluating the effect of each loss term, and comparing with advanced baselines (e.g. with scene graphs [7], [8]).

REFERENCES

[1] P. Anderson, A. Chang, D. S. Chaplot, A. Dosovitskiy, S. Gupta, V. Koltun, J. Kosecka, J. Malik, R. Mottaghi, M. Savva, and A. R. Zamir, "On evaluation of embodied navigation agents," 2018. [Online]. Available: https://arxiv.org/abs/1807.06757

[2] W. Yang, X. Wang, A. Farhadi, A. Gupta, and R. Mottaghi, "Visual semantic navigation using scene priors," 2018. [Online]. Available: https://arxiv.org/abs/1810.06543

[3] D. S. Chaplot, D. P. Gandhi, A. Gupta, and R. R. Salakhutdinov, "Object goal navigation using goal-oriented semantic exploration," in *Advances in Neural Information Processing Systems*, 2020, pp. 4247–4258.

[4] A. Rosinol, A. Violette, M. Abate, N. Hughes, Y. Chang, J. Shi, A. Gupta, and L. Carlone, "Kimera: From slam to spatial perception with 3d dynamic scene graphs," *The International Journal of Robotics Research*, vol. 40, no. 12-14, pp. 1510–1546, 2021.

[5] H. Bavle, J. L. Sanchez-Lopez, M. Shaheer, J. Civera, and H. Voos, "S-graphs+: Real-time localization and mapping leveraging hierarchical representations," *IEEE Robotics and Automation Letters*, vol. 8, no. 8, pp. 4927–4934, 2023.

[6] N. Hughes, Y. Chang, S. Hu, R. Talak, R. Abdulhai, J. Strader, and L. Carlone, "Foundations of spatial perception for robotics: Hierarchical representations and real-time systems," *The International Journal of Robotics Research*, vol. 43, no. 10, pp. 1457–1505, 2024.

[7] Z. Ravichandran, L. Peng, N. Hughes, J. D. Griffith, and L. Carlone, "Hierarchical representations and explicit memory: Learning effective navigation policies on 3d scene graphs using graph neural networks," in *IEEE International Conference on Robotics and Automation*, 2022, pp. 9272–9279.

[8] P. Kremer, H. Bavle, J. L. Sanchez-Lopez, and H. Voos, "S-nav: Semantic-geometric planning for mobile robots," *arXiv preprint arXiv:2307.01613*, 2023.

[9] A. T. Baisch and R. J. Wood, "Design and fabrication of the harvard ambulatory micro-robot," in *Robotics Research: The 14th International Symposium ISRR*. Springer, 2011, pp. 715–730.

[10] M. Egerstedt, J. N. Pauli, G. Notomista, and S. Hutchinson, "Robot ecology: Constraint-based control design for long duration autonomy," *Annual Reviews in Control*, vol. 46, pp. 1–7, 2018.

[11] R. Bormann, F. Jordan, W. Li, J. Hampp, and M. Hägele, "Room segmentation: Survey, implementation, and analysis," in *IEEE International Conference on Robotics and Automation*, 2016, pp. 1019–1026.

[12] J. Jung, C. Stachniss, and C. Kim, "Automatic room segmentation of 3d laser data using morphological processing," *ISPRS International Journal of Geo-Information*, vol. 6, no. 7, p. 206, 2017.

[13] A. Kleiner, R. Baravalle, A. Kolling, P. Pilotti, and M. Munich, "A solution to room-by-room coverage for autonomous cleaning robots," in *IEEE International Conference on Intelligent Robots and Systems*, 2017, pp. 5346–5352.

[14] M. Hiller, C. Qiu, F. Particke, C. Hofmann, and J. Thielecke, "Learning topometric semantic maps from occupancy grids," in *IEEE International Conference on Intelligent Robots and Systems*, 2019, pp. 4190–4197.

[15] M. Luperto, T. P. Kucner, A. Tassi, M. Magnusson, and F. Amigoni, "Robust structure identification and room segmentation of cluttered indoor environments from occupancy grid maps," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7974–7981, 2022.

[16] Z. Wang, D. Threatt, S. B. Andersson, and R. Tron, "Do more with less: Single-model, multi-goal architectures for resource-constrained robots," in *IEEE International Conference on Intelligent Robots and Systems*, 2023, pp. 1940–1946.

[17] Z. Li, J. D. Wegner, and A. Lucchi, "Topological map extraction from overhead images," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 1715–1724.

[18] H. Ling, J. Gao, A. Kar, W. Chen, and S. Fidler, "Fast interactive object annotation with curve-gcn," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 5257–5266.

[19] S. Wei and S. Ji, "Graph convolutional networks for the automated production of building vector maps from aerial images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–11, 2021.

[20] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European conference on computer vision*. Springer, 2020, pp. 213–229.

[21] Y. Hu, Z. Wang, Z. Huang, and Y. Liu, "Polybuilding: Polygon transformer for end-to-end building extraction," *arXiv preprint arXiv:2211.01589*, 2022.

[22] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97.'Towards New Computational Principles for Robotics and Automation'*, 1997, pp. 146–151.

[23] M. Luperto, L. Fochetta, and F. Amigoni, "Exploration of indoor environments through predicting the layout of partially observed rooms," in *International Conference on Autonomous Agents and MultiAgent Systems*, 2021, p. 836–843.

[24] R. Shrestha, F.-P. Tian, W. Feng, P. Tan, and R. Vaughan, "Learned map prediction for enhanced mobile robot exploration," in *IEEE International Conference on Robotics and Automation*, 2019, pp. 1197–1204.

[25] G. J. Stein, C. Bradley, and N. Roy, "Learning over subgoals for efficient navigation of structured, unknown environments," in *Conference on robot learning*. PMLR, 2018, pp. 213–222.

[26] H. Li, Q. Zhang, and D. Zhao, "Deep reinforcement learning-based automatic exploration for navigation in unknown environment," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 6, pp. 2064–2076, 2020.

[27] R. I. Arnob and G. J. Stein, "Improving reliable navigation under uncertainty via predictions informed by non-local information," in *IEEE International Conference on Intelligent Robots and Systems*, 2023, pp. 2830–2837.

[28] T. P. Kucner, M. Luperto, S. Lowry, M. Magnusson, and A. J. Lilienthal, "Robust frequency-based structure extraction," in *IEEE International Conference on Robotics and Automation*, 2021, pp. 1715–1721.

[29] J. Illingworth and J. Kittler, "A survey of the hough transform," *Computer vision, graphics, and image processing*, vol. 44, no. 1, pp. 87–116, 1988.

[30] W. Wu, X.-M. Fu, R. Tang, Y. Wang, Y.-H. Qi, and L. Liu, "Data-driven interior plan generation for residential buildings," *ACM Transactions on Graphics*, vol. 38, no. 6, nov 2019.

[31] G. Borgefors, "Distance transformations in digital images," *Computer vision, graphics, and image processing*, vol. 34, no. 3, pp. 344–371, 1986.

[32] T. Li, D. Ho, C. Li, D. Zhu, C. Wang, and M. Q.-H. Meng, "Houseexpo: A large-scale 2d indoor layout dataset for learning-based algorithms on mobile robots," in *IEEE International Conference on Intelligent Robots and Systems*, 2020, pp. 5839–5846.

[33] H. Umari and S. Mukhopadhyay, "Autonomous robotic exploration based on multiple rapidly-exploring randomized trees," in *IEEE International Conference on Intelligent Robots and Systems*, 2017, pp. 1396–1402.

[34] R. Shrestha, F.-P. Tian, W. Feng, P. Tan, and R. Vaughan, "Learned map prediction for enhanced mobile robot exploration," in *IEEE International Conference on Robotics and Automation*, 2019, pp. 1197–1204.