2025 IEEE/SICE International Symposium on System Integration (SII)
January 21-24, 2025, Munich, Germany

Fast LiDAR Informed Visual Search in Unseen Indoor Environments

Ryan Gupta[1], Kyle Morgenstein[1], Steven Ortega[2] and Luis Sentis[1]

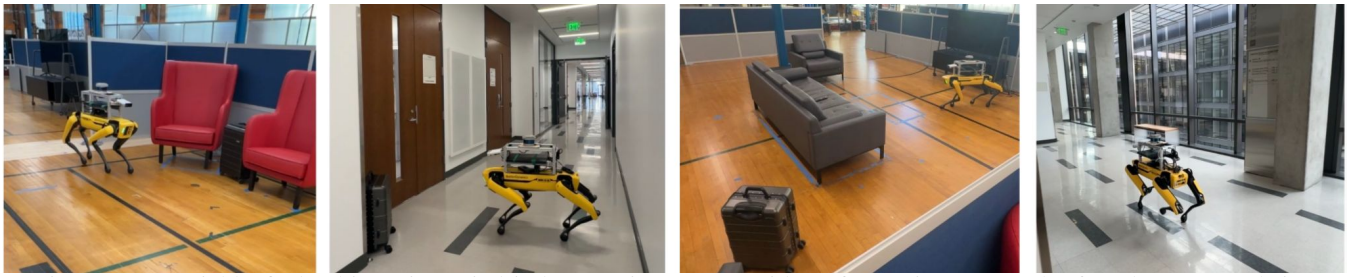

Fig. 1: Spot quickly finds objects in varied indoor environments. Video of experiments can be found on our website.

***Abstract*— This paper details a system for fast visual exploration and search without prior map information. We leverage frontier based planning with both LiDAR and visual sensing and augment it with a perception module that contextually labels points in the surroundings from wide Field of View 2D LiDAR scans. The goal of the perception module is to recognize surrounding points more likely to be the search target in order to provide an informed prior on which to plan next best viewpoints. The robust map-free scan classifier used to label pixels in the robot's surroundings is trained from expert data collected using a simple cart platform equipped with a map-based classifier. We propose a novel utility function that accounts for the contextual data found from the classifier. The resulting viewpoints encourage the robot to explore points unlikely to be permanent in the environment, leading the robot to locate objects of interest faster than several existing baseline algorithms. Our proposed system is further validated in real-world search experiments for single and multiple search objects with a Spot robot in two unseen environments. Videos of experiments, implementation details and open source code can be found at `https://sites.google.com/view/lives-2024/home`.**

I. INTRODUCTION

Planning and real-world execution for autonomous visual exploration are receiving significant attention from the robotics community due to their relevance in a wide range of scenarios including inspection [1], search [2], disaster response [3], reconnaissance and surveillance [4]. As mobile robots begin to proliferate in our communities, visual planning capabilities remain fundamental for autonomous agents to achieve success in human environments [5]. Active planners in unknown spaces are required to make decisions based on incomplete and noisy information about partial environments. This requires planners equipped to generate high quality plans under uncertainty.

Today's robots are frequently equipped with LiDAR sensors that cast a full view of the surroundings. State of the art visual exploration planners like [6, 7] demonstrate efficiency by fusing LiDAR and visual information. At the same time, the potential to detect additional information in LiDAR scans for online control has grown with advances in parallel computing. While methods exist for 3D LiDAR segmentation [8, 9], they are limited by high computational requirements [10]. In contrast, projection-based methods run online but are used to detect moving obstacles for autonomous vehicles [11, 12]. The literature also demonstrates planning under reduced uncertainty in unseen environments by exploiting features common to the places they are designed for [13, 14]. Despite improvements in efficiency of planning algorithms for both LiDAR and vision sensors [6], it remains an open problem to enhance performance in autonomous planning and execution [15]. To this end, we propose to exploit features common to many indoor environments using a lightweight scan segmentation method for efficient visual search planning.

The key insight in this work is to exploit contextual information available in wide Field of View LiDAR scans to filter out points unlikely to be the search target. This deviates from traditional exploration methods, which focus on completeness. Instead, the proposed method prioritizes regions more likely to be the search target in the robot's surroundings. We hypothesize exploring agents can improve time efficiency in visual search by exploiting this additional information. A map based classifier is used to train a model to recognize features inherent to many indoor environments. The perception model feeds the planner the non-permanent points, *non-map points*. These points guide the Next Best View (NBV) planner towards possible search target locations, assumed to be in the set of non-map points in the environment. Examples of search environments are shown in Fig. 1.

The contributions of this work can be summarized as follows:

- Achieve significantly faster search indoors than baselines by augmenting multisensor exploration with contextual information from LiDAR scans
- A scan segmentation policy that identifies non-permanent points in unseen, real-world environments
- A novel frontier-based exploration formulation that accounts for segmented scan information

[1]Department of Aerospace Engineering and Engineering Mechanics, University of Texas at Austin, Austin, TX 78712 USA `ryan.gupta@utexas.edu`
[2]Department of Mechanical Engineering, University of Texas at Austin, Austin, TX 78712 USA

II. Related Works

The two common approaches to exploration include frontier [16] and information-theoretic sampling methods [17]. Sampling approaches avoid the expensive frontier generation process but require evaluation of every sample, motivating hybrid approaches [18]. Alternatively, [19] use RRTs to search for frontier points; achieving efficiency using multiple trees and decoupling tree expansion from movement. Another approach expands RRTs with accessible space and executes the edge with the highest information gain in a receding horizon fashion [20]. The key difference between these methods and ours is the addition of segmented scans, which is shown to significantly improve task completion time.

State of the art multisensor exploration is shown with an underwater vehicle [7] and multiple UAVs [6]. They combine frontier generation for two sensors and use a single utility function to score frontier candidates for surface inspection. We modify their method to sample candidates *at* frontier centroids instead of *surrounding* them. This modified method is equivalent to our proposed planner *without* the use of segmented scan information and represents a third baseline method. This baseline clearly demonstrates the positive effect of the proposed contextual information.

Partially observable markov decision processes (POMDP) are commonly used in visual object search [21, 5]. The latter presents a platform-generalizable 3D viewpoint planner, capable of reasoning about viewpoints underneath furniture or between appliances. Their work includes a 2D planner for multi-room search, however, it is not tailored to search larger spaces for things that may be in plain sight; a skill that offers real-world application in search and rescue. Because our goals involve searching larger spaces quickly, the aforementioned exploration algorithms provide more fair baseline comparisons for the proposed system.

World prediction is used to learn a model of relevant environments and exploit their structure to explore high-value regions first. Ref. [13] use topological features like loops or dead-ends from a database of subterranean tunnels to inform a frontier based exploration policy. Instead, [14] considers occupancy anticipation from egocentric RGB-D for navigation by an exploration planner. Ref. [22] learns to predict locations of exits in building with a convolutional neural network using a database of building blueprints. Our proposed system leverages a similar principle to exploit the structures inherent to many indoor environments - like planar walls, hallways, rooms and loops - in order to guide the planner. We achieve this in a novel way by segmenting scans to identify non-permanent features in the environment. Ultimately, the addition of these features are shown to enable our method to find objects of interest more quickly than the examined baselines.

III. Methods

The overall scan classifier and planning module are described in Alg. 1 with key components detailed in this section.

Algorithm 1 Map-Free LiDAR Informed Search()

Input: s_i, x_i
LiDAR Scan, Robot Pose
Output: x^*_{next} (Next viewpoint)
$s_i^{\text{class}} = \text{NeuralNetworkClassifier}(s_i)$
$\mathcal{M}_i^{\text{LiDAR}} \leftarrow \text{LiDARMapUpdate}(s_i, x_i, \mathcal{M}_{i-1}^{\text{LiDAR}})$
$\mathcal{M}_i^{\text{Visual}} \leftarrow \text{VisualMapUpdate}(s_i, x_i, \mathcal{M}_{i-1}^{\text{Visual}})$
$\{\mathbb{C}^{\text{LiDAR}}\} = \text{GetLiDARFrontiers}(\mathcal{M}_i^{\text{LiDAR}})$
$\{\mathbb{C}^{\text{Visual}}\} = \text{GetVisualFrontiers}(\mathcal{M}_i^{\text{Visual}})$
for $n \leftarrow 1$ to N **do** ▷ for each frontier c_n in $\mathbb{C}$
  for $j \leftarrow 1$ to 4 **do** ▷ for each viewpoint at c_n
    $\text{util}_{n,j} = \text{ComputeSampleUtility}(c_n, x_i, s_i^{\text{class}})$
  end for
end for
$x^*_{\text{next}} \leftarrow \arg\max_{n,j}(\text{util}_{n,j})$
return x^*_{next}

A. Environment

The environment discretized into a set of cells $\mathcal{E}$, split into two subsets $\mathcal{E}^{\text{free}} \subset \mathcal{E}$ and $\mathcal{E}^{\text{occ}} \subset \mathcal{E}$, representing free and occupied cells. Both sets are initially unknown to the robot, but are assumed to follow a structure inherent to indoor environments (e.g. hallways, planar walls, open rooms and loops).

B. Ground Truth LiDAR Scan Classification

Ground truth classification uses map $\mathcal{M}$ to divide $\mathcal{E}^{\text{occ}}$ into $\mathcal{E}^{\text{non-map}}$ and $\mathcal{E}^{\text{map}}$. In this work, $\mathcal{E}^{\text{map}}$ are points in the environment that are deemed to be Long-Term Features (LTFs) during classification, while $\mathcal{E}^{\text{non-map}}$ are points that are either Short-Term Features (STFs) or Dynamic Features (DFs) [23]. LTFs represent permanent features while STFs and DFs are non-permanent static and dynamic points, respectively. Given $\mathcal{M}$, represented as a set of lines $\{l_i\}_{1:n}$, points are classified into these categories as follows.

Let x_i denote robot pose, and s_i denote observation at time step t_i. Observation s_i consists of n_i 2D points, $s_i = \{p_i^j\}_{j=1:n_i}$. Observations are transformed from local to global frame using affine transformation $T_i \in SE(3)$.

1) LTF: First, an analytic ray cast is performed [24] to determine expected laserscan based on map $\mathcal{M}$ and current robot position x_i. Given observations, the probability that points correspond to the static map can be written:

$$P(p_i^j | x_i, M) = \exp\left(-\frac{\text{dist}(T_i p_i^j, l_j)^2}{\Sigma_s}\right) \tag{1}$$

where Σ_s is the scalar variance of observations from sensor accuracy. If Eq. 1 is above a threshold, point p_i^j is an LTF.

2) STF: Remaining points are either STFs or DFs. Observations at current time i, p_i^j, are compared with prior observations at time k, p_k^l to determine correspondence between points in subsequent observations. The likelihood of a point corresponding between time steps is computed as:

$$P(p_i^j, p_k^l | x_i, x_k) = \exp\left(-\frac{||T_i p_i^j - T_k p_k^l||^2}{\Sigma_s}\right) \tag{2}$$

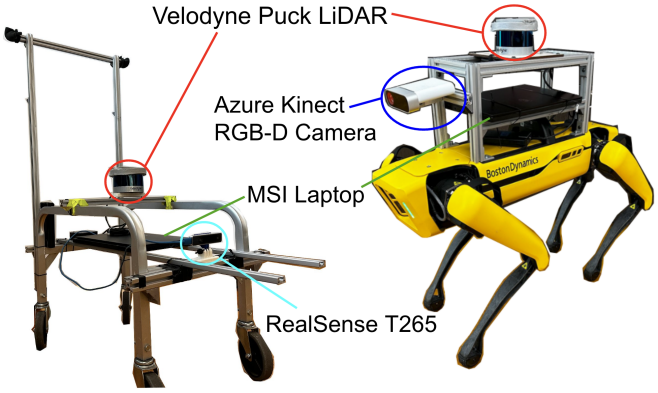

Fig. 2: The cart used for labeled data acquisition and the Spot robot used for deployment. The RealSense provides odometry estimate to the cart, required for ground-truth estimation. Spot is equipped with an RGB-D Azure Kinect for detection.

where p_k^l is the nearest point from p_i^j among points which does not belong to LTF at other timesteps, defined as

$$p_k^l = \arg\min ||T_i p_i^j - T_k p_k^l|| \tag{3}$$

When Eq. 2 is above a threshold, point p_i^j is classified as an STF. Remaining points in p_i^j are classified as DFs. The result is classified scan s_i^{class}. Open source code is published by Biswas et. al. [23] at [25].

C. Map-free LiDAR Scan Classification

The classification method defined in III-B is limited by the necessity of static map information $\mathcal{M}$. Therefore, this work uses supervised learning to train a model to reproduce binary classifications for $\{p_i^j\}_{j=1:n_i}$ in scan s_i, where $\{p_i^j\}_{j=1:n_i,class} \in \{-1, 1\}$

1) Dataset and Data Acquisition: We first collect a dataset, however data acquisition with a robot is difficult to scale due to its mechanical limits. To ease data acquisition we design an easily maneuverable cart platform (see fig. 2) to mimic the robot. Data is labeled and recorded online as the cart is pushed through eight indoor environments on the university campus that vary in size and shape in attempt to capture generic features found in different indoor environments. Training and test maps can be found on the website. A laptop with static map information on board the cart receives odometry and scan data from sensors, then uses the method described in the previous section to maintain a pose estimate and collect dataset $\mathcal{D}$, consisting of tuples of data. The tuples are given as $\mathcal{D} = \{(x_i, s_i, s_i^{\text{class}})\}_{i=1}^N$, where x_i represents robot pose, s_i is the raw LiDAR scan data, and s_i^{class} is the classified scan. Note that non-map elements in s_i^{class} are value -1 and map elements are value 1. N is the total number of data points collected.

The Velodyne Puck has $n_i = 897$ and classified scans are recorded at 5Hz. A dataset of size $N \simeq 145,000$ is collected in this study, representing roughly 8 hours of time. Training data can be found on the project website.

2) Architecture and Training: In practice, the model only receives raw scan data, pose estimates and previously *predicted* labels from the robot during operation and must classify pixel-wise the scan online. As a result, we formulate pixel-wise classification as a supervised learning problem during training and an auto-regressive problem during inference. Given dataset $\mathcal{D}$, we shuffle and batch the dataset between time steps and locations to minimize location-based and temporal bias during training. Because no map information is provided explicitly, we provide the model with a history buffer of k time steps for scan ranges and robot poses, and $k-1$ previously estimated ground truth labels. We corrupt the ground truth labels during training by randomly bit-flipping the classification of 10% of the labels, chosen by sampling indices from a uniform random distribution. In order to produce a k length estimated label history input, we take the exponential weighted average of the $k-1$ corrupted pixel-wise classifications to estimate the current timestep's classification.

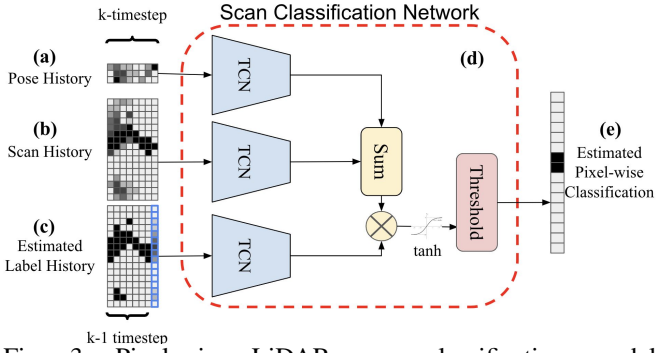

Fig. 3: Pixel-wise LiDAR scan classification model architecture used to speed up search by providing information to the planner. k is the length of the history buffer. (a) $[3, k]$ pose history matrix containing $[x, y, \theta]$. (b) $[n_i, k]$ LiDAR range history matrix. (c) $[n_i, k-1] \cup [n_i, 1]$ estimated label history matrix concatenated with its pixel-wise exponential weighted average. (d) The model consists of three temporal-convolutional encoders (TCN) (i.e. pose encoder, scan encoder, and label encoder). The encoded poses, scans, and labels are combined to produce a pixel-wise classification of the LiDAR scan. In (e) a threshold (positive/negative) is applied to the raw logits such that each pixel is classified as either a map point or a non-map point.

During inference, the label history buffer is populated by previous estimates from the policy, cropped by a length $k-1$ sliding-window. The history buffer is concatenated with the updated pixel-wise exponential weighted average in the same way as was done during training to obtain the k length estimated label history input. At initialization, the policy is bootstrapped with zero-value poses, scans, and labels, and run for k steps of inference until the history buffer is full. Code for training a model can be found on the project website.

The full model architecture is shown in Fig. 3 and consists of a temporal-convolutional encoder (TCN) for each input. Each TCN contains a single convolutional layer with scan-wise circular padding and a single linear layer. The scan and label encoders have kernel size $[k, k]$, and the pose encoder has kernel size $[1, 3]$. Hyperbolic tangent activations are used for all layers. The output of each TCN is the same

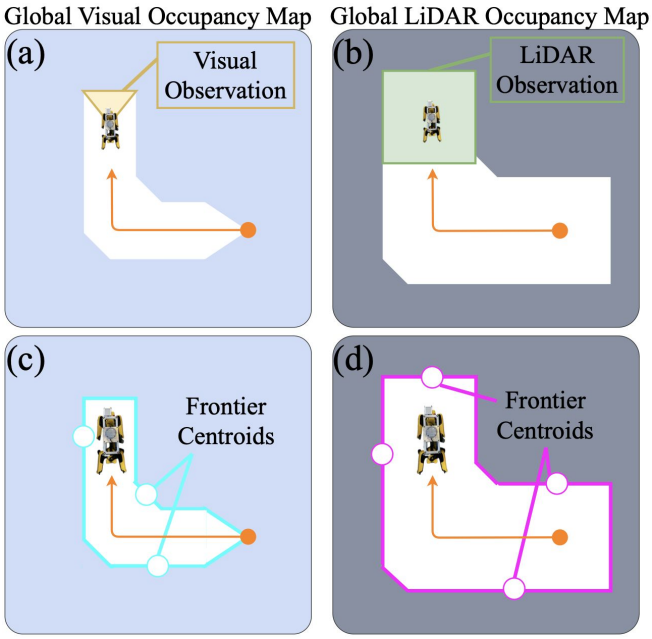

Fig. 4: Map updates are performed from observation at each timestep for visual (a) and LiDAR (b) sensors. White cells are known free and blue/grey cells are unknown. Next, frontiers and their centroids are computed for each of the two sensors (c) and (d).

size as the LiDAR scan to be classified ($n_i = 897$). The output of the pose history TCN is summed with the output of the scan encoder as a pose correction before element-wise multiplication with the label encoding, which may flip the sign of the term. The result is normalized by applying a hyperbolic tangent function such that each value is $\{-1, 1\}$. A threshold (positive/negative) is applied to the raw logits such that each pixel is classified as either a map point or a non-map point. The policy is trained with mean squared error loss against ground truth classifications. Training for 20 epochs on a workstation Nvidia 3080 12Gb takes 3 minutes.

D. Map Updates

Global occupancy maps (search maps) are maintained for both the LiDAR and visual sensor (see fig. 4(a-b)). Global search maps are implemented as occupancy grids. They are updated at each step from local sensor occupancy grids, which are implemented using the ROS Costmap2D package [26]. All cells in both global maps are initialized to be unknown, $e = -1 \; \forall e \in \mathcal{E}$. The robot is equipped with a visual sensor with a cone shaped FoV, implemented as a triangular costmap. The LiDAR sensor has a 360 degree FoV, represented as a square-type costmap. Observations for both sensors are a set of range estimates to the nearest occupied point $e \in \mathcal{E}^{\text{occ}}$. Points between the robot and the nearest occupied point along each ray of the LiDAR scan are free points $e \in \mathcal{E}^{\text{free}}$. Given incoming observations from the robot, each global costmap is updated as free space $e = 0$ or occupied $e = 1$.

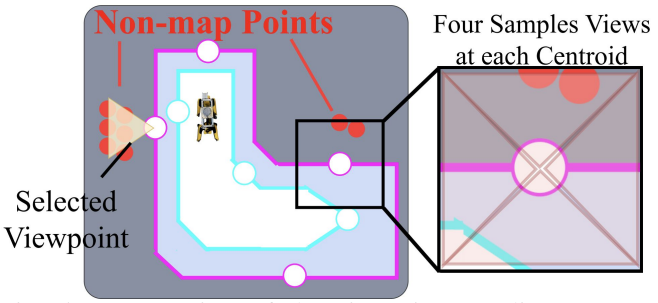

Fig. 5: An overview of the viewpoint sampling process. Four viewpoints are considered at each centroid, shown as red triangles. The highest scoring viewpoint is shown as a yellow triangle to be sent to the robot.

E. Viewpoint Planning

The planner sends the viewpoint at each planning step that results in the highest likelihood to find the target of interest. This involves picking viewpoints that balance exploration of unknown space in the environment with exploiting non-map points from LiDAR scans. This is achieved in several steps: candidate viewpoints are generated, viewpoints are scored, and best sample is fed to the robot.

1) Candidate Viewpoint Generation: Frontier points in each of the two global search maps are generated using a process similar to [16], shown in Fig. 4(c-d). Any free cell ($e = 0$) adjacent to an unknown cell ($e = -1$) is a frontier edge, which are grouped together into frontiers. Any group of frontier points beyond a minimum size becomes its own frontier. Frontier centroids of each frontier in both the visual and LiDAR search maps are found by clustering. Four candidate viewpoints are evaluated at each frontier centroid to account for the limited FoV of the vision sensor (see fig. 5).

2) Viewpoint Selection: Candidate viewpoints are scored using the following utility function. High value viewpoints are those that are near the robot, allow inspection of many unknown map points and result in inspection of non-map points. The utility function is a weighted sum of the following: (1) Penalize distance from robot position to viewpoint, (2) Reward expected number of unknown cells discovered, (3) Reward viewpoints with frontiers near the path to the candidate viewpoint, and (4) Reward viewpoints that result in inspection of non-map points. Scaling parameters on each term are tuned during experiments. Weights are set such that the agent is inclined to prioritize non-map points and those viewpoints that result in high number of expected cells discovered. The next waypoint is selected by sending the highest utility viewpoint, x^*_{next}, to the robot.

IV. Results

We test three hypotheses through our evaluation: (1) The proposed approach to inform an exploring agent with non-map features is effective at finding search targets, (2) it reduces detection time compared against existing algorithms, and (3) it can be deployed successfully on a real robot in unseen indoor environments.

Map Name	Target Difficulty	NBVP [20]	RRT [19]	MFE [7]	Ground Truth Classification	Our Method
Apartment	Easy	$112 \pm 17.9(70\%)$	$72 \pm 16.9(80\%)$	**36 ± 4.4 (100%)**	**34 ± 3.3 (100%)**	**34 ± 4.9 (100%)**
	Hard	$177 \pm 11.7(30\%)$	$128 \pm 13.3(60\%)$	**106 ± 18.2 (100%)**	**92 ± 8.6(100%)**	**90 ± 10.6(100%)**
Office	Easy	$70 \pm 13.9(80\%)$	$54 \pm 9.4(80\%)$	**34 ± 7.7 (100%)**	**26 ± 8.7 (100%)**	**24 ± 6.9 (100%)**
	Hard	$300 \pm 0(0\%)$	**111 ± 25.1 (80%)**	$141 \pm 14.8(90\%)$	**99 ± 23.7 (100%)**	**105 ± 34.5(100%)**

TABLE I: Average time in seconds with standard deviation (and success rate) to find the object of interest in simulation.

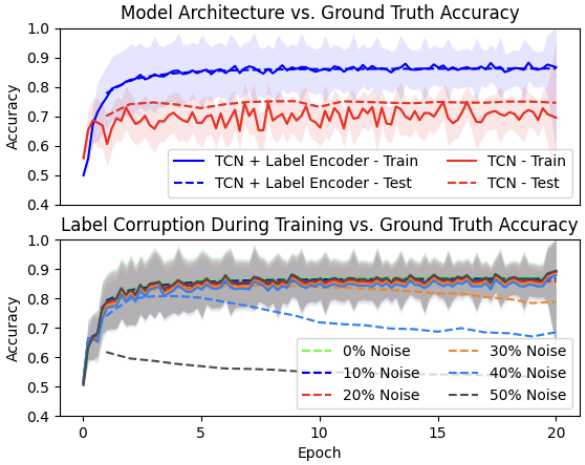

Fig. 6: Ablation studies over policy architecture (top) and injected noise during training (bottom). The inclusion of the label history buffer yields 11.63% higher test accuracy. The policy is robust up to 30% to bit-flipping errors in the label history buffer. The mean accuracy depicts the 5-step moving average.

A. LiDAR Scan Classification

We compare the learned classification policy to the ground truth classifier. The output of the policy is a vector of size n_i normalized between -1 and 1 for each scan point. If a value is non-negative, the point belongs to $\mathcal{E}^{\text{map}}$. Otherwise the point belongs to $\mathcal{E}^{\text{non-map}}$. We express performance in terms of per-scan accuracy of the classified LiDAR scans from the policy vs. ground truth from map information. This metric for classified scan s_i^{class} at time i compared to ground truth $s_i^{\text{class, true}}$ is given by

$$\text{accuracy} = \frac{1}{n_i}\sum_{j=0}^{n_i}\{+1 \text{ if } p_i^j == p_i^{j,\text{true}} \text{ else } 0\} \quad (4)$$

The final test accuracy of the policy compared to ground truth is $86.19\% \pm 0.03\%$ with no a priori map requirement. We perform two ablation studies, modulating the model architecture and injected noise during training, with results in Fig. 6. The inclusion of the label encoder is responsible for a 11.63% increase in test accuracy. Because the robot moves relatively slowly compared to the policy update frequency, recently predicted labels strongly bias the current estimate toward the correct classification. We fix the history buffer to 9 time steps, corresponding to 1.8 seconds. The history buffer enables the policy to reason about its environment even though no reference map is provided. During training, uniform random noise is injected into the label history buffer to simulate inference-time auto-regressive accumulated errors. The policy is robust up to 30% of injected noise before the policy begins to over-fit and performance degrades. We select a policy trained with 10% injected noise to deploy on the robot. All ablation studies were performed with a fixed seed (0) and initialized with identical weights. We do not tune the random seed.

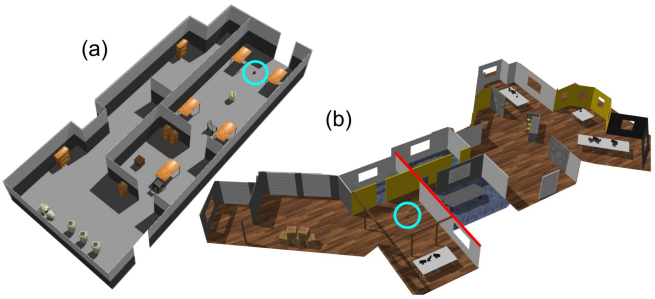

Fig. 7: Environments used for baseline comparison. Robot start pose is denoted by a cyan circle. (a) The Apartment (20x30m) simulation environment in the hard configuration. Easy configuration has fewer objects. (b) The Office (25x45m) environment [27]. Easy setup involves the search target to the left of the red line and hard involves the search target on the right.

B. Simulation and Baseline Comparison

Simulations are performed in Gazebo with the Turtlebot and ROS navigation. Baselines are: 1) modified Multisensor Frontier Exploration (MFE) as described in section II [7], 2) Next-Best View Planner (NBVP) [20], 3) Rapidly-Exploring Random Tree (RRT) [19], and 4) our method, but informed with ground-truth scan classifications as described in section III-B. Two Gazebo worlds are tested with easy and hard setups in each (fig. 7). In all settings there was one search object, the Clearpath Husky. The apartment (20x30m) examines the impact of non-target objects on the planner, the easy setting with fewer objects. The office (25x45m) tests the method in a large environment under significantly varied target position. Easy and hard settings are defined by the search object in the same or different half of the environment as the robot starting position. Both difficulty levels in each map are repeated 10 times. Table I reports the results. The time limit given to planners is 2 and 3 minutes in the apartment easy and hard settings then 3 and 5 minutes in the office easy and hard settings. Trials where the planner fails or the time limit is reached are reported as a failure and failed trials are reported as the maximum allotted time.

Overall, the results support the proposed approach, reporting 100% success and outperforming baselines across all evaluation settings. Further, the nearly identical performance between the proposed approach with both predicted and ground truth labels support the efficacy of the perception module. While MFE performed similarly to the proposed approach in the easy apartment, the lack of the labeled scan data slowed task completion by 18-42% across

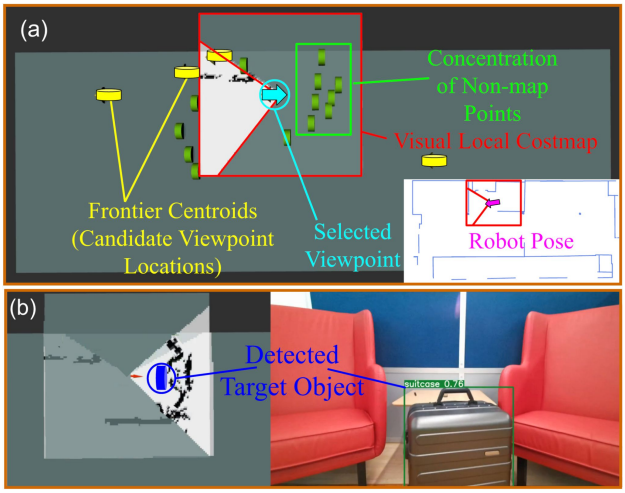

Fig. 8: An example experimental scenario. (a) The first planning step. Candidate viewpoints are sampled at yellow disks. The robot chooses the next viewpoint at the shown cyan arrow in order to visually inspect the concentration of non-map points behind the robot's starting pose. The robot pose on the ground truth map is shown in the lower right (unknown to the robot, not included in the training dataset). (b) The concentration of non-map points is found to be two red chairs and the search object (suitcase). This result reinforces that the scan classifier successfully recognizes non-permanent features in unseen indoor environments.

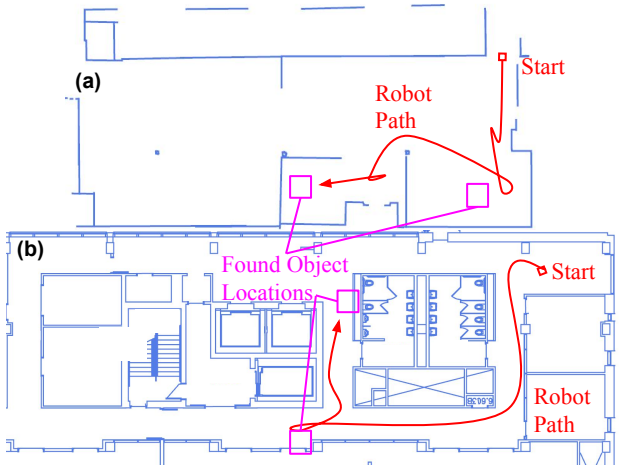

Fig. 9: Paths the robot took to find the objects of interest (suitcases) in the video scenarios 2 and 3. (a) Two search objects in the 20x30m apartment environment. (b) Two search objects in the 15x38m hallway environment.

the remaining settings. NBVP and RRT baselines struggled in most settings, with the former averaging about 50% success. RRT outperformed other baselines in the hard office environment, however, was 20% less successful and moderately slower than the proposed approach. The overall results support our method, suggesting the benefits of focusing on non-map points. This focus allows the agent to move on when the target is unlikely to be nearby. These results support hypotheses (1) and (2) that contextual LiDAR information significantly improves performance in the search task.

C. Hardware Experiments

Experiments are in a 20mx30m apartment and a 15mx38m hallway environment. These two environments are not included in dataset $\mathcal{D}$ used to train the classification policy. A Boston Dynamics Spot is equipped with a Velodyne Puck LiDAR and Azure Kinect RGB-D camera. We first evaluate the accuracy of the trained classifier in these out of distribution environments to ground truth information. This is accomplished by teleoperating the robot while running the ground truth and map-free classifiers in parallel. The accuracy in the two out of distribution environments are 84.0168% in the apartment over a 105 second deployment and 84.869% in the hallway over a 230 second deployment. These suggests the classifier's ability to distinguish permanent features common to many indoor environments.

During experiments, YOLOv5 [28] is used for detection. Objects are localized by first converting PointCloud data into the optical frame and then converting to pixel coordinates using camera intrinsics. Bounding boxes are then used to obtain the depth information from coinciding pixels. A simple experiment is shown in Fig. 8 and demonstrates the benefits of non-map information in guiding the vision sensor in the robot's surroundings. Instead of selecting based on number of unknown cells, contextual information leads the agent to inspect many non-map features nearby, resulting in detection. Videos of experimentation, implementation details, code, training data, and all maps can be found on the project website `https://sites.google.com/view/lives-2024/home`. The video shows two experiments in the apartment and one in the hallway. Fig. 9 demonstrates the path of the robot in scenarios 2 and 3. In both apartment scenarios, the robot tends to look towards furniture like couches, shelves and tables. Similarly, Spot also tends to inspect trash cans and other objects in the hallway. This behavior demonstrates the impact of non-map points and further supports the efficacy of the segmentation policy. In the hallway environment the robot is guided by non-map points at several key forks, finding both search objects (suitcase) in the environment. After first inspecting the concentration of non-map points situated behind the robot, the decisions the robot makes at the two key forks it intersects lead it to quickly find the two suitcases in the environment. Overall, the results of hardware experiments on the Spot support hypothesis (3).

V. Conclusions

We introduce a novel visual planner that leverages contextual information found from wide field-of-view LiDAR scans. The map-free LiDAR classifier identifies non-permanent points in the environment for the planner's attention. The method is shown to outperform baselines by 10-30% in simulation. Through ablative studies, we validate our architecture and training approach for map-free scan classification. Experiments confirm the effectiveness of the visual search planner and learning method in real-world scenarios. Future work plans to leverage the non-map points to generate higher quality maps and expand to multi-robot applications.

VI. Acknowledgments

This research was supported in part by NSF Award #2219236 (GCR: Community Embedded Robotics: Understanding Sociotechnical Interactions with Long-term Autonomous Deployments) and Living and Working with Robots, a core research project of Good Systems, a UT Grand Challenge. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

References

[1] LV Nguyen, TH Le, and Quang Phuc Ha. "Grey Wolf Optimization-Based Path Planning for Unmanned Aerial Vehicles in Bridge Inspection". In: *2024 IEEE/SICE International Symposium on System Integration (SII)*. IEEE. 2024, pp. 810–815.

[2] Xuning Chen, Jianying Zheng, and Qinglei Hu. "A Hybrid Planning Method for 3D Autonomous Exploration in Unknown Environments With a UAV". In: *IEEE Transactions on Automation Science and Engineering* (2023).

[3] Qingchen Bi, Xuebo Zhang, Jian Wen, Zhangchao Pan, Shiyong Zhang, Runhua Wang, and Jing Yuan. "CURE: A Hierarchical Framework for Multi-Robot Autonomous Exploration Inspired by Centroids of Unknown Regions". In: *IEEE Transactions on Automation Science and Engineering* 99 (2023), pp. 1–14.

[4] Li Liang, Fang Deng, Jianan Wang, Maobin Lu, and Jie Chen. "A reconnaissance penetration game with territorial-constrained defender". In: *IEEE Transactions on Automatic Control* 67.11 (2022), pp. 6295–6302.

[5] Kaiyu Zheng, Anirudha Paul, and Stefanie Tellex. "A System for Generalized 3D Multi-Object Search". In: *arXiv preprint arXiv:2303.03178* (2023).

[6] Graeme Best, Rohit Garg, John Keller, Geoffrey A Hollinger, and Sebastian Scherer. "Resilient multi-sensor exploration of multifarious environments with a team of aerial robots". In: *Robotics: Science and Systems (RSS)*. 2022.

[7] Eduard Vidal, Narcís Palomeras, Klemen Istenič, Nuno Gracias, and Marc Carreras. "Multisensor online 3D view planning for autonomous underwater exploration". In: *Journal of Field Robotics* 37.6 (2020), pp. 1123–1147.

[8] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. "Pointnet++: Deep hierarchical feature learning on point sets in a metric space". In: *Advances in neural information processing systems* 30 (2017).

[9] Maciej Zamorski, Maciej Zieba, Piotr Klukowski, Rafal Nowak, Karol Kurach, Wojciech Stokowiec, and Tomasz Trzciński. "Adversarial autoencoders for compact representations of 3D point clouds". In: *Computer Vision and Image Understanding* 193 (2020), p. 102921.

[10] Alok Jhaldiyal and Navendu Chaudhary. "Semantic segmentation of 3D LiDAR data using deep learning: a review of projection-based methods". In: *Applied Intelligence* 53.6 (2023), pp. 6844–6855.

[11] Xieyuanli Chen, Benedikt Mersch, Lucas Nunes, Rodrigo Marcuzzi, Ignacio Vizzo, Jens Behley, and Cyrill Stachniss. "Automatic labeling to generate training data for online LiDAR-based moving object segmentation". In: *IEEE Robotics and Automation Letters* 7.3 (2022), pp. 6107–6114.

[12] Benedikt Mersch, Xieyuanli Chen, Ignacio Vizzo, Lucas Nunes, Jens Behley, and Cyrill Stachniss. "Receding moving object segmentation in 3d lidar data using sparse 4d convolutions". In: *IEEE Robotics and Automation Letters* 7.3 (2022), pp. 7503–7510.

[13] Manish Saroya, Graeme Best, and Geoffrey A Hollinger. "Online exploration of tunnel networks leveraging topological CNN-based world predictions". In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2020, pp. 6038–6045.

[14] Santhosh K Ramakrishnan, Ziad Al-Halah, and Kristen Grauman. "Occupancy anticipation for efficient exploration and navigation". In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*. Springer. 2020, pp. 400–418.

[15] Chaoqun Wang, Wenzheng Chi, Yuxiang Sun, and Max Q-H Meng. "Autonomous robotic exploration by incremental road map construction". In: *IEEE Transactions on Automation Science and Engineering* 16.4 (2019), pp. 1720–1731.

[16] Brian 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'*. IEEE. 1997, pp. 146–151.

[17] Benjamin Charrow, Sikang Liu, Vijay Kumar, and Nathan Michael. "Information-theoretic mapping using cauchy-schwarz quadratic mutual information". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2015, pp. 4791–4798.

[18] Anna Dai, Sotiris Papatheodorou, Nils Funk, Dimos Tzoumanikas, and Stefan Leutenegger. "Fast frontier-based information-driven autonomous exploration with an mav". In: *2020 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2020, pp. 9570–9576.

[19] Hassan Umari and Shayok Mukhopadhyay. "Autonomous robotic exploration based on multiple rapidly-exploring randomized trees". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 1396–1402.

[20] Andreas Bircher, Mina Kamel, Kostas Alexis, Helen Oleynikova, and Roland Siegwart. "Receding horizon "next-best-view" planner for 3D exploration". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2016, pp. 1462–1468.

[21] Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. "Planning and acting in partially observable stochastic domains". In: *Artificial intelligence* 101.1-2 (1998), pp. 99–134.

[22] Jeffrey A Caley, Nicholas RJ Lawrance, and Geoffrey A Hollinger. "Deep learning of structured environments for robot search". In: *Autonomous Robots* 43 (2019), pp. 1695–1714.

[23] Joydeep Biswas and Manuela M Veloso. "Episodic non-markov localization". In: *Robotics and Autonomous Systems* 87 (2017), pp. 162–176.

[24] Joydeep Biswas and Manuela Veloso. "Depth camera based indoor mobile robot localization and navigation". In: *2012 IEEE International Conference on Robotics and Automation*. IEEE. 2012, pp. 1697–1702.

[25] Joydeep Biswas. URL: https://github.com/ut-amrl/enml.

[26] Eitan Marder-Eppstein, David V. Lu, and Dave Hershberger. 2024. URL: http://wiki.ros.org/costmap_2d.

[27] Clearpath Robotics. URL: https://github.com/clearpathrobotics/cpr_gazebo/tree/noetic-devel.

[28] Glenn Jocher. *YOLOv5 by Ultralytics*. Version 7.0. May 2020. DOI: 10.5281/zenodo.3908559. URL: https://github.com/ultralytics/yolov5.