Information-theoretic Abstraction of Semantic Octree Models for Integrated Perception and Planning

Daniel T. Larsson[1]   Arash Asgharivaskasi[2]   Jaein Lim[1]   Nikolay Atanasov[2]   Panagiotis Tsiotras[1]

Abstract— In this paper, we develop an approach that enables autonomous robots to build and compress semantic environment representations from point-cloud data. Our approach builds a three-dimensional, semantic tree representation of the environment from raw sensor data which is then compressed by a novel information-theoretic tree-pruning approach. The proposed approach is probabilistic and incorporates the uncertainty in semantic classification inherent in real-world environments. Moreover, our approach allows robots to prioritize individual semantic classes when generating the compressed trees, so as to design multi-resolution representations that retain the relevant semantic information while simultaneously discarding unwanted semantic categories. We demonstrate the approach by compressing semantic octree models of a large outdoor, semantically rich, real-world environment. In addition, we show how the octree abstractions can be used to create semantically-informed graphs for motion planning, and provide a comparison of our approach with uninformed graph construction methods such as Halton sequences.

I. INTRODUCTION

Dense volumetric environment representations such as occupancy grid maps [1], [2], multi-resolution hierarchical models [3], [4], and signed distance fields (SDF) [5], [6], provide valuable information to both human and autonomous robots, as evidenced by their utility in search and rescue [7], safe navigation [8], and terrain modeling [9]. Moreover, the inclusion of semantic information, such as in metric-semantic SLAM methods of [10]–[12], allows robots to build more sophisticated world models by affording autonomous systems the ability to not only discern occupied from free space, but to also distinguish between the types of objects in their surroundings. Recent work has leveraged Bayesian statistics to develop algorithms that build semantic environment representations, encoding categorical (semantic) information using probabilistic methods which naturally capture the uncertainty robots hold regarding their world [10], [11]. These models supply autonomous robots an abundance of information, enabling them to intelligently reason about their surroundings with details such as the location, geometry and size of obstacles, or the presence of humans, cars, and other semantic information.

While constructing environment models is an important step for mobile robot autonomy, sensors provide overabun-

dant, and often redundant, information for a specific task. It is therefore of interest to not only *build* environment models but to also *compress* them to form abstracted representations, allowing robots to focus their (possibly scarce) resources on the relevant aspects of the operating domain. The use of abstractions in the form of multi-resolution environment model compressions have seen widespread deployment in the autonomous systems community. Examples include [13]–[18], where abstractions are leveraged in order to alleviate the computational cost of planning and decision making in both single and multi-robot applications. Abstractions have also been utilized to reduce the memory required to store environment representations [19], [20] and to alleviate the computational complexity of evaluating cost functions in active-sensing applications [21]. However, while identifying the relevant aspects of a problem to generate task-relevant abstractions has long been considered vital to intelligent reasoning [22]–[27], the means by which they are generated has traditionally been heavily reliant on user-provided rules.

To this end, a number of studies have considered the generation of task-relevant abstractions for control and decision-making that model relevant information via the statistics of the process. Examples of such works include [28]–[30], where ideas from information theory, specifically rate-distortion [31] and the information bottleneck method [32], are employed to develop approaches that identify and preserve task-relevant information by modeling the relevant information as a random variable that is correlated with the source (i.e., the original representation). While the above studies consider frameworks that form abstractions that preserve relevant information, they do not result in representations of any particular structure (e.g., quadtrees, octrees). To address this issue, the work of [33], [34] uncovered connections between hierarchical tree structures and signal encoders to formulate an information-theoretic compression problem that allows optimal task-relevant tree abstractions to be obtained as a solution to an optimization problem. Moreover, extensions of the tree abstraction problem from [33] that consider generating tree abstractions in the presence of both relevant and irrelevant information sources has recently appeared in the literature [35]. Importantly, the frameworks developed in [33]–[35] require minimal input from the system designer, enabling robots to generate compressed tree representations of their world that are driven by task-specific information.

The goal of this paper is to bridge the gap between map building and abstraction construction by developing a framework that performs both tasks simultaneously. The proposed approach employs an information-theoretic tree compression

method to find provably optimal tree abstractions of large environments that both retain information regarding task-relevant semantic classes and remove those that are considered task-irrelevant. We demonstrate our approach in a real-world outdoor environment, and show how the framework can be employed to generate semantically-informed (colored) graphs to reduce the computational effort required for motion planning.

II. PROBLEM STATEMENT

Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space. The *Shannon entropy* [36, p. 14] of a discrete random variable $X : \Omega \rightarrow \mathbb{R}$ with probability mass function $p(x) = \mathbb{P}\{\omega \in \Omega : X(\omega) = x\}$ is denoted $H(X)$.[1] Provided two distributions $p(x)$ and $\bar{p}(x)$ over the same set of outcomes, the *Kullback-Leibler divergence* [36, p. 19] is $\mathrm{D}_{\mathrm{KL}}(p(x), \bar{p}(x)) = \sum_x p(x) \log[p(x)/\bar{p}(x)]$. Given a collection of distributions $p_1(x), \ldots, p_\ell(x)$ over the same set of outcomes, the *Jensen-Shannon divergence* [37] with respect to the weights $\Pi \in \mathbb{R}_+^\ell$ is given by $\mathrm{JS}_\Pi(p_1(x), \ldots, p_\ell(x)) = \sum_{i=1}^\ell \Pi_i \mathrm{D}_{\mathrm{KL}}(p_i(x), \bar{p}(x))$, where $\bar{p}(x) = \sum_{i=1}^\ell \Pi_i p_i(x)$, $0 \leq \Pi_i \leq 1$ for all $1 \leq i \leq \ell$ and $\sum_{i=1}^\ell \Pi_i = 1$.

We assume a grid-world representation of the environment $\mathcal{W} \subset \mathbb{R}^3$, where each cell contains semantic information regarding one or more of K possible semantic classes contained in the set $\mathcal{K} = \{0, 1, \ldots, K\}$. In the sequel, we let the semantic class $0 \in \mathcal{K}$ denote free space and each $k \in \mathcal{K} \setminus \{0\}$ represent a distinct semantic category (e.g., building, vegetation, road, etc). A hierarchical, three-dimensional (3-D) multi-resolution octree representation of $\mathcal{W}$ is a tree[2] $\mathcal{T}$ consisting of a set of nodes $\mathcal{N}(\mathcal{T})$ and edges $\mathcal{E}(\mathcal{T})$ that describe the node interconnections, where each non-leaf node in the tree has exactly 8 children. We denote the set of children of any node $n \in \mathcal{N}(\mathcal{T})$ by $\mathcal{C}(n)$, the leaf nodes by $\mathcal{N}_{\mathrm{leaf}}(\mathcal{T})$, and the interior nodes by $\mathcal{N}_{\mathrm{int}}(\mathcal{T})$ [35]. Lastly, we let $\mathcal{T}_\mathcal{W}$ denote the finest-resolution octree representation of $\mathcal{W}$; that is, $\mathcal{T}_\mathcal{W}$ is the octree whose leaf nodes coincide with the unit cells of $\mathcal{W}$.

Our goal is to develop a perception and abstraction approach that allows for semantic octree representations to be built from sensor data while simultaneously being optimally compressed in a low-cardinality tree data structure. For this, we require two components: (i) the source (i.e., the quantity to be compressed); and (ii) any relevant or irrelevant information that is to be retained or removed, respectively. To this end, the source, relevant and irrelevant information are represented by the random variables $X : \Omega \rightarrow \mathcal{N}_{\mathrm{leaf}}(\mathcal{T}_\mathcal{W})$ with distribution $p(x)$ (i.e., the outcomes x of X are the finest-resolution cells), $Y_i : \Omega \rightarrow \{0, 1\}$, $i \in \mathcal{K}_Y$ and $Z_j : \Omega \rightarrow \{0, 1\}$, $j \in \mathcal{K}_Z$, respectively, where the sets $\mathcal{K}_Y$ and $\mathcal{K}_Z$ are subsets of $\mathcal{K}$ that contain the indices of the relevant and irrelevant semantic classes. The relationship between source, relevant and irrelevant information is specified by $p(x, y_{1:|\mathcal{K}_Y|}, z_{1:|\mathcal{K}_Z|})$. We consider the following problem.

Semantic Octree Building-Compression: Given an octree representation $\mathcal{T}_\mathcal{W}$ and the joint probability distribution $p(x, y_{1:|\mathcal{K}_Y|}, z_{1:|\mathcal{K}_Z|})$ from perceptual data, find a compressed multi-resolution octree $\mathcal{T}$ from $\mathcal{T}_\mathcal{W}$ by solving the problem:

$$\max_{\mathcal{T} \in \mathcal{T}^\mathcal{O}} \sum_{i \in \mathcal{K}_Y} \beta_i I_{Y_i}(\mathcal{T}) - \sum_{j \in \mathcal{K}_Z} \gamma_j I_{Z_j}(\mathcal{T}) - \alpha I_X(\mathcal{T}), \quad (1)$$

where $\mathcal{T}^\mathcal{O}$ is the space of all octree representations of $\mathcal{W}$, $\beta \in \mathbb{R}_+^{|\mathcal{K}_Y|}$, $\gamma \in \mathbb{R}_+^{|\mathcal{K}_Z|}$ and $\alpha \in \mathbb{R}_+$ specify the relative importance of relevant information retention, irrelevant information removal, and compression, respectively, and the functions $I_{Y_i} : \mathcal{T}^\mathcal{O} \rightarrow \mathbb{R}_+$, $I_{Z_j} : \mathcal{T}^\mathcal{O} \rightarrow \mathbb{R}_+$ and $I_X : \mathcal{T}^\mathcal{O} \rightarrow \mathbb{R}_+$ quantify the amount of relevant, irrelevant and compression information contained in the octree (see [35, p. 9]). Note that $p(x, y_{1:|\mathcal{K}_Y|}, z_{1:|\mathcal{K}_Z|})$ enters into (1) via $I_{Y_i}(\mathcal{T})$, $I_{Z_j}(\mathcal{T})$ and $I_X(\mathcal{T})$.

III. INFORMATION-THEORETIC ABSTRACTION OF SEMANTIC OCTREES

We first provide some theoretical background on tree compression. We will employ the G-tree search algorithm [35] to solve the compression problem in (1). The goal of the G-tree search algorithm is to find a compressed representation $N : \Omega \rightarrow \mathcal{N}_{\mathrm{leaf}}(\mathcal{T})$, $\mathcal{T} \in \mathcal{T}^\mathcal{O}$, of the source X (i.e., the leaf cells of $\mathcal{T}_\mathcal{W}$) in the form of an octree $\mathcal{T}$ of $\mathcal{W}$ according to (1), where the distribution $p(n)$ is related to the source according to:

$$p(n) = \sum_{x \in \mathcal{N}_{\mathrm{leaf}}(\mathcal{T}_{\mathcal{W}(n)})} p(x), \quad (2)$$

and $\mathcal{N}_{\mathrm{leaf}}(\mathcal{T}_{\mathcal{W}(n)}) \subseteq \mathcal{N}_{\mathrm{leaf}}(\mathcal{T}_\mathcal{W})$ are the leaf nodes of the subtree of $\mathcal{T}_\mathcal{W}$ rooted node $n \in \mathcal{N}_{\mathrm{leaf}}(\mathcal{T})$. Since the octree solution $\mathcal{T} \in \mathcal{T}^\mathcal{O}$ to (1) is not known a-priori, we may compute the value of $p(n)$ for all nodes $n \in \mathcal{N}(\mathcal{T}_\mathcal{W})$ recursively according to

$$p(n) = \sum_{n' \in \mathcal{C}(n)} p(n'). \quad (3)$$

The objective of G-tree search is to determine which nodes of the original octree $\mathcal{T}_\mathcal{W}$ should be leaf nodes of the compressed representation $\mathcal{T}$. To accomplish its goal, G-tree search exploits the structure of problem (1) to devise a node pruning rule, called the G-function, defined by:

$$G(n; \beta, \gamma, \alpha) = \max\{\Delta J(n; \beta, \gamma, \alpha) + \sum_{n' \in \mathcal{C}(n)} G(n'; \beta, \gamma, \alpha), \ 0\}, \quad (4)$$

if $n \in \mathcal{N}_{\mathrm{int}}(\mathcal{T}_\mathcal{W})$ and $p(n) > 0$, and $G(n; \beta, \gamma, \alpha) = 0$ otherwise. The function $\Delta J(n; \beta, \gamma, \alpha)$ is the one-step reward for expanding node $n \in \mathcal{N}_{\mathrm{int}}(\mathcal{T}_\mathcal{W})$ and is given by

$$\Delta J(n; \beta, \gamma, \alpha) = \sum_{i \in \mathcal{K}_Y} \beta_i \Delta I_{Y_i}(n) - \sum_{j \in \mathcal{K}_Z} \gamma_j \Delta I_{Z_j}(n) - \alpha \Delta I_X(n), \quad (5)$$

---

[1] if the distribution $p(x)$ is understood from context we may write $H(p)$ in place of $H(X)$.
[2] a tree is an a-cyclic connected graph [38].

where the functions $\Delta I_{Y_i}(n)$, $\Delta I_{Z_j}(n)$ and $\Delta I_X(n)$ quantify the incremental amount of relevant, irrelevant and compression information, respectively, contributed by the node $n \in \mathcal{N}_{\text{int}}(\mathcal{T}_\mathcal{W})$. These functions are, in turn, defined by:

$$\Delta I_{Y_i}(n) = p(n)\text{JS}_\Pi(p(y_i|n_1'), \ldots, p(y_i|n_{|\mathcal{C}(n)|}')), \quad (6)$$

$$\Delta I_{Z_j}(n) = p(n)\text{JS}_\Pi(p(z_j|n_1'), \ldots, p(z_j|n_{|\mathcal{C}(n)|}')), \quad (7)$$

$$\Delta I_X(n) = p(n)H(\Pi), \quad (8)$$

where $p(y_i|n_u')$ for $n_u' \in \mathcal{C}(n)$ are recursively computed via

$$p(y_i|n_u') = \sum_{n'' \in \mathcal{C}(n_u')} \Pi_{n''} p(y_i|n''), \ i \in \mathcal{K}_Y, \quad (9)$$

$$p(z_j|n_u') = \sum_{n'' \in \mathcal{C}(n_u')} \Pi_{n''} p(z_j|n''), \ j \in \mathcal{K}_Z, \quad (10)$$

and $\Pi \in \mathbb{R}_+^{|\mathcal{C}(n_u')|}$ has entries $\Pi_{n''} = p(n'')/p(n_u')$. Once the G-values (4) are known from an inverse breadth-first node traversal of $\mathcal{T}_\mathcal{W}$, G-tree search considers nodes $n \in \mathcal{N}_{\text{int}}(\mathcal{T}_\mathcal{W})$ in a top-down manner to determine whether or not they should be part of the solution to (1). See [35] for more details regarding the G-tree search algorithm.

Careful inspection of (4) and (6)-(8) reveals that G-tree search depends on $p(x, y_{1:|\mathcal{K}_Y|}, z_{1:|\mathcal{K}_Z|})$ only via the distributions $p(y_i|x)$, $p(z_j|x)$ and $p(x)$. Thus, we assume that a distribution $p(x)$ over leaf nodes is provided, and determine $p(y_i|x)$ and $p(z_j|x)$ from semantic octree perception data. Our solution consists of two phases: (i) *the update pass*: inserts or updates nodes in the current octree based on semantic perception data, and (ii) *the octree compression pass*: executes G-tree search to compress the environment representation that is built as part of phase 1. Next, we describe the tree-building process before delineating how $p(y_i|x)$ and $p(z_j|x)$ are obtained from perception data.

A. Updating Tree Nodes from Perceptual Data

We adopt the semantic model proposed in [11] to build a hierarchical Bayesian multi-class octree representation of the world. The tree-building algorithm builds the finest resolution octree $\mathcal{T}_\mathcal{W}$ from observations (see Fig. 1), and maintains a truncated probability distribution over semantic classes, represented by a random variable $S : \Omega \to \mathcal{K}$ for each leaf node $x \in \mathcal{N}_{\text{leaf}}(\mathcal{T}_\mathcal{W})$. In more detail, given a node $x \in \mathcal{N}_{\text{leaf}}(\mathcal{T}_\mathcal{W})$, if $k \in \mathcal{K}_3(x) \cup \{0\}$ then $p(S = k|x)$ is provided by the octree, where $\mathcal{K}_3(x)$ is the set of three most likely classes of node x. The octree also stores a forth entry, corresponding to $p(S \in \mathcal{K} \setminus (\mathcal{K}_3(x) \cup \{0\})|x)$, which is the probability of the event that the node x belongs to a semantic class other than the three most likely or free space. Furthermore, to reduce the memory required to store the map, the algorithm will prune nodes whose children all have identical multi-class probability distributions.

B. Extracting Semantic Information from the Octree Data Structure

For any $i \in \mathcal{K}_Y$, the conditional distribution $p(y_i|x)$ is derived from the semantic octree according to $p(y_i = 1|x) = p(S = i|x)$, with an analogous expression for $p(z_j|x)$, $j \in \mathcal{K}_Z$. In practice, obtaining the conditional distribution $p(y_i|x)$
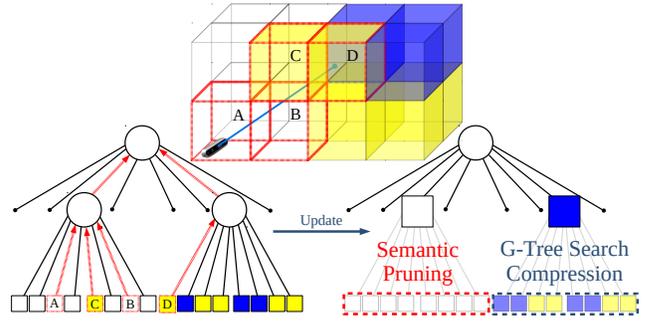

Fig. 1: Semantic octree update from a new observation indicating cell D is blue and C is free space. Each leaf node with color-encoded object class is depicted as a square, inner nodes are represented by circles, and unexplored cells are shown as dots. Multi-class probabilities and G-values along the paths from leaf nodes to the root, highlighted in red, need to be updated due to the new perceptual data.

(resp. $p(z_j|x)$) presents a challenge, since the tree-building algorithm only maintains the truncated semantic probabilities. Thus, we cannot directly obtain $p(y_i|x)$ or $p(z_j|x)$ from the semantic octree structure, as the three most likely semantic classes may differ from node to node. Instead, we must generate the distribution $p(s|x)$ over *all* semantic classes. To this end, for each leaf node $x \in \mathcal{N}_{\text{leaf}}(\mathcal{T}_\mathcal{W})$, we collect the truncated semantic probabilities and uniformly distribute the probability $p(S \in \mathcal{K} \setminus (\mathcal{K}_3(x) \cup \{0\})|x)$ among the outstanding classes. Once the distributions $p(y_i|x)$, $p(z_j|x)$, and $p(x)$ are known for the leaf nodes of the octree $\mathcal{T}_\mathcal{W}$, we may apply the recursive relations (3) and (9)-(10) to update the semantic distributions for all interior nodes.

It is worth mentioning that each node $n \in \mathcal{N}_{\text{int}}(\mathcal{T}_\mathcal{W})$ may not always have a full set of children since the tree-building algorithm instantiates nodes only for the observed locations in the environment. To see why missing children pose a challenge, assume node $n \in \mathcal{N}_{\text{int}}(\mathcal{T}_\mathcal{W})$ has only a single child; that is $\mathcal{C}(n) = \{n'\}$. From (3) and (6)-(10), we see that if $\mathcal{C}(n) = \{n'\}$, then we have $\Delta I_X(n) = 0$, $\Delta I_{Y_i}(n) = 0$ and $\Delta I_{Z_j}(n) = 0$; so no information is lost in aggregating the child node n' to its parent n. In order to remedy this issue, we account for absent children by instantiating a maximum entropy distribution (i.e., uniform over $\mathcal{K}$) and a value of $p(n)$ for nodes not represented in the octree, from which $p(y_i|x)$ and $p(z_j|x)$ are obtained analogously to the existing leaf nodes. Importantly, the recursive structure of (3) and (9)-(10) implies that only $p(y_i|n')$, $p(z_j|n')$ and $p(n')$ for missing child nodes n' are required to compute the values of their parent n, thereby alleviating the need to insert the missing nodes in the octree structure.

C. Updating G-values from Local Tree Information

From (9)-(10) we note that computing $p(y_i|n)$ and $p(z_j|n)$ for any $n \in \mathcal{N}_{\text{int}}(\mathcal{T}_\mathcal{W})$ can be done from knowledge of $p(y_i|n')$ and $p(z_j|n')$ for $n' \in \mathcal{C}(n)$. Moreover, these updates do not require $p(x)$ to be a valid probability distribution, since (9)-(10) depend only the *relative* weights $p(n')/p(n)$. However, this structure is not shared by the G-function (4),

Algorithm 1: Joint semantic-tree construction and compression

---

input : Semantic point cloud $\mathcal{P}$, G-tree search weights $(\beta, \gamma, \alpha) \in \mathbb{R}_+^{|\mathcal{K}_Y|} \times \mathbb{R}_+^{|\mathcal{K}_Z|} \times \mathbb{R}_+$.

output: Compressed octree representation $\mathcal{T}^*$ of $\mathcal{W}$.

---

1 **if** *point cloud data recieved* **then**
2     $x \leftarrow \texttt{createOrUpdateNode}(\mathcal{P})$;
3     $n \leftarrow x$;
4     $G_\Pi(n; \beta, \gamma, \alpha) = 0$;
5   **while** *Parent(n)* $\neq \varnothing$ **do**
6        $\bar{n} \leftarrow \texttt{Parent}(n)$;
7   **if** $\mathcal{C}(\bar{n}) \subseteq \mathcal{N}_{leaf}(\mathcal{T}_\mathcal{W})$ **then**
8           $(p(y_i|\bar{n}), p(z_j|\bar{n}), p(\bar{n})) \leftarrow \texttt{getDist}(\bar{n})$;
9   **else**
10           $(p(y_i|\bar{n}), p(z_j|\bar{n}), p(\bar{n})) \leftarrow \texttt{chdDist}(\bar{n})$;
11        $G_\Pi(\bar{n}; \beta, \gamma, \alpha) \leftarrow \texttt{updateGvals}(\beta, \gamma, \alpha)$;
12        $n \leftarrow \bar{n}$;
13     $\mathcal{T}^* \leftarrow \texttt{GtreeSearch}(n_\mathsf{R})$;
14 **return** $\mathcal{T}^*$

---

since the latter has an explicit dependence on $p(x)$. It is therefore of interest to investigate if characteristics of the G-function, or some equivalent, can be expressed in terms of only relative weights. To this end, we define $G_\Pi : \mathcal{N}(\mathcal{T}_\mathcal{W}) \times \mathbb{R}_+^{|\mathcal{K}_Y|} \times \mathbb{R}_+^{|\mathcal{K}_Z|} \times \mathbb{R}_+ \to \mathbb{R}_+$ according to: if $n \in \mathcal{N}_{\text{int}}(\mathcal{T}_\mathcal{W})$ and $p(n) > 0$ then

$$
\begin{aligned}
&G_\Pi(n; \beta, \gamma, \alpha) \\
&= \max\{ \sum_{i \in \mathcal{K}_Y} \beta_i \text{JS}_\Pi^{Y_i}(n) - \sum_{j \in \mathcal{K}_Z} \gamma_i \text{JS}_\Pi^{Z_j}(n) - \alpha H(\Pi) \\
&\qquad + \sum_{n' \in \mathcal{C}(n)} \Pi_{n'} G_\Pi(n'; \beta, \gamma, \alpha),\ 0\}, \quad (11)
\end{aligned}
$$

and $G_\Pi(n; \beta, \gamma, \alpha) = 0$ otherwise, where for $i \in \mathcal{K}_Y$, $\text{JS}_\Pi^{Y_i}(n) = \text{JS}_\Pi(p(y_i|n_1'), \dots, p(y_i|n_{|\mathcal{C}(n)|}'),\ n_u' \in \mathcal{C}(n)$ with $\text{JS}_\Pi^{Z_j}(n)$ defined analogously. This brings us to the following result.

Proposition 3.1: Let $n \in \mathcal{N}(\mathcal{T}_\mathcal{W})$. Then $G_\Pi(n; \beta, \gamma, \alpha) > 0$ if and only if $G(n; \beta, \gamma, \alpha) > 0$.

    Proof: The proof is given in the Appendix. ∎
As a result of Proposition 3.1, we may predicate our pruning on the function $G_\Pi(n; \beta, \gamma, \alpha)$ in place of $G(n; \beta, \gamma, \alpha)$ without sacrificing any of the theoretical guarantees of the G-tree search method. In the next section, we present the joint tree-building and compression algorithm.

D. The Joint Tree-Building and Compression Algorithm

The joint semantic octree-building and compression framework is shown in Algorithm 1. The update process (i.e., lines 1-12) is triggered by the availability of new semantic point-cloud data. Moreover, the function $\texttt{createOrUpdateNode}(\mathcal{P})$ either creates or updates the semantic information of an existing leaf node of the octree $\mathcal{T}_\mathcal{W}$ from the new data (see Fig. 1). Since incoming information is always inserted as a leaf of $\mathcal{T}_\mathcal{W}$, we, in line 4, set

the leaf-node condition for the G-values. We then traverse the octree bottom-up in lines 5-12, visiting the sequence of node parents until the root node is reached, updating G-values and semantic distributions along the way (see Fig. 1). If the node $\bar{n}$ is a parent of a leaf node, such that line 7 is true, then we extract the full semantic distribution as detailed by Section III-B for the children $x \in \mathcal{C}(\bar{n})$. This is done by the routine $\texttt{getDist}(\bar{n})$, before applying the recursive updates (9)-(10). In contrast, if $\bar{n}$ is not a parent of a leaf, then $p(y_i|n')$ and $p(z_j|n')$ are known for all $n' \in \mathcal{C}(\bar{n})$, and thus, $\texttt{chdDist}(\bar{n})$ retrieves the distributions of the children and applies (9)-(10), before updating the G-values in line 11 according to (11). We then call the tree-compression algorithm in line 13 by passing the root node of $\mathcal{T}_\mathcal{W}$, denoted n_R, to G-tree search.

In the language employed at the start of this section, lines 2-12 comprise the update pass (phase 1) and line 13 constitutes the octree compression step (phase 2). Note that the bottom-up recursion defined by lines 5-12 of Algorithm 1 is made possible by the following two observations. First, at each time instance for which point cloud information is available, the semantic tree-building algorithm inserts (or updates) exactly one leaf node of the current octree representation of $\mathcal{W}$. Secondly, the function $G_\Pi(n; \beta, \gamma, \alpha)$ and distributions $p(y_i|n)$, $p(z_j|n)$ and $p(n)$ can all be updated from, and only depend on, immediate child information. Thus, the nodes traversed by the recursion in lines 5-12 of Algorithm 1 are precisely those whose G-values and semantic distributions are effected by the new perceptual data received (see Fig. 1).

IV. REAL-WORLD EXPERIMENTS

In this section, we present and discuss the results obtained from field-test experiments utilizing our proposed tree compression pipeline, illustrated in Fig. 2. The outdoor environment contains a number of elevation changes and a combination of urban and rural elements. The FCHarD-Net classifier [39] pre-trained on RUGD dataset [40] is employed for semantic classification, providing 24 possible categorizations (i.e., $\mathcal{K} = \{0, \dots, 24\}$). The output of classification is combined with range sensing to produce range-category observations in the form of semantically-annotated 3-D point clouds, which are used to estimate the finest-resolution semantic octree $\mathcal{T}_\mathcal{W}$ (see Fig. 2c). Finally, the compressed multi-resolution octree is obtained via the framework presented in Section III. Results are presented for two scenarios: (i) to study the semantic tree-building and compression algorithm detailed in Section III, and (ii) to demonstrate how the abstractions can be employed in semantically-informed planning problems.

A. Semantic Perception and Tree Compression

Figure 3 shows the normalized degree of information retained regarding each of the visible semantic classes of the environment in Fig. 2, as a function of the G-tree search weights (solution number) depicted in Fig. 4. By comparing Figs. 3 and 4 we make a few observations. First, we observe that solution 5 considers all semantic classes as relevant, and thus the abstraction returned by G-tree search retains all information. Secondly, we see the impact of the
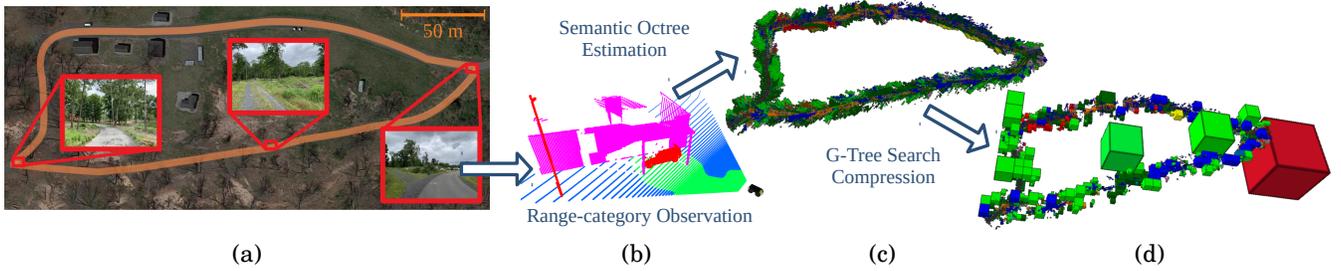
Fig. 2: Summary of the semantic octree building and compression framework. (a) The robot navigates in a large, semantically-rich, outdoor environment. (b) Range-category observations are gathered to build a semantic octree of the environment (c) that is then compressed by G-tree search (d). While shown here sequentially, the semantic octree estimation and G-tree search compression may run concurrently as described in Section III. Colors in the trees (c)-(d) correspond with semantic classes.
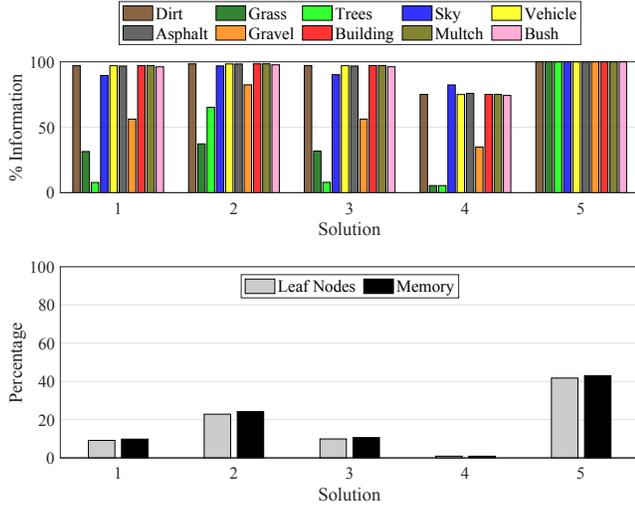

Fig. 3: (top) Percentage of information retained for each semantic class as a function of the G-tree search weights (solution number) with respect to the corresponding quantities in the finest-resolution tree $\mathcal{T}_{\mathcal{W}}$. (bottom) Percentage of leaf nodes and approximate memory required to store the compressed tree relative to the corresponding values of the finest resolution octree $\mathcal{T}_{\mathcal{W}}$. Memory results assume each tree node requires 256 bits of memory.
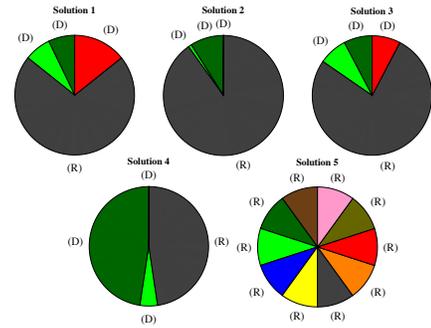

Fig. 4: Weights for G-tree search. Colors correspond with the semantic class coloring, (R) is a relevant class, and (D) is an irrelevant one. In solutions 2 and 4, the (D) near the top corresponds to the red class (building). Note that classes marked (R) have weights β_i, whereas those marked (D) have weights γ_j in (1). Classes not shown have weight zero.

weights on the octree solution by considering solutions 2 and 4. To this end, note that solution 4 contains considerably less information regarding the tree and grass classes as compared with solution 2, which occurs since the 4th solution penalizes the retention of both grass and trees (dark and light green, respectively) to a much higher degree than solution 2. Notice also that solution 4 contains less information regarding *all* classes compared with solution 2, since the priority of information removal outweighs the importance of information retention resulting with a more compressed octree, confirmed by Fig. 3(bottom). Moreover, observe that the G-tree search method is able to find a compressed octree that retains most of the semantic information, as seen by solution 5. To understand why this is the case, recall that node information $\Delta I_{Y_i}(n)$ and $\Delta I_{Z_j}(n)$ in the G-tree

search method is quantified by the JS-divergence. Thus, nodes with smaller values of $\Delta I_{Y_i}(n)$ and $\Delta I_{Z_j}(n)$ imply that the semantic distributions $p(y_i|n)$ and $p(z_j|n)$ are more similar to the distributions $p(y_i|n')$ and $p(z_j|n')$ of their children $n' \in \mathcal{C}(n)$ as compared with nodes having greater values of $\Delta I_{Y_i}(n)$ and $\Delta I_{Z_j}(n)$. Contrast this with the ad-hoc pruning rule employed by the tree-building process of Section III-A (see Fig. 1), which prunes nodes based on individual elements of the distributions $p(y_i|n)$ and $p(z_j|n)$. Consequently, it may happen that nodes are not pruned by the ad-hoc pruning rule but contain little to no information due to the similarity of the semantic probability distributions of its immediate children. The G-tree search method is able to exploit these redundancies, leading to the high degrees of compression seen in Fig. 3.

B. Semantic Perception, Abstraction and Planning

Lastly, we discuss how the perception-abstraction framework developed in Section III can be employed to construct a more (semantically) informed graph than conventional techniques (e.g., Halton sequence [41]) for use in colored graph-search planning algorithms. We consider the Class-Ordered A* (COA*) algorithm [42], which computes a semantically-informed path and allows for both desired (i.e.,
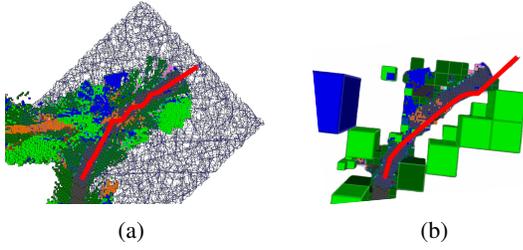
Fig. 5: (a) The optimal path found by COA* in the graph constructed with a Halton sequence. (b) The optimal path found by COA* in the graph constructed based on the compressed octomap nodes.

relevant) and undesired semantic classes to be specified. The COA* algorithm searches a weighted colored (semantic) graph to find the shortest path that contains the least number of edges in unwanted classes [42]. Vertices in the search graph correspond to a two-dimensional coordinate and heading configuration of a non-holonomic ground robot, that is (x, y, θ), and edges represent the Reeds-Shepp curve [43]. Thus, all paths in the graph are dynamically feasible. Graph vertices are classified (i.e., given a color) according to the semantic information contained in the (semantic) octree node corresponding to the volumetric region containing the (x, y)-coordinate which the vertex represents.

Shown in Fig. 5 are example paths obtained from employing COA* on colored (semantic) graphs when paved road is considered a relevant (preferred) class for planning. For this planning scenario, we utilize our framework to generate the semantically-informed colored graph shown in Fig. 5(b) by specifying the asphalt (paved road) class as relevant, and those of grass and trees as irrelevant, for G-tree search. As a result, we see in Fig. 5(b) that areas containing paved road retain relatively high resolution as compared with areas that contain little to no paved road and a greater amount of grass and trees. Contrast these observations with the graph generated from a Halton sequence in Fig. 5(a), which is agnostic to the semantic information and contains many vertices that correspond to irrelevant planning classes. To provide quantitative results, we averaged planning results over 50 search instances for both methods. From our study, we observed that the optimal path was found about 10% percent faster on average in the semantically informed (compressed) graph generated by the framework from Section III. Moreover, we noted that the standard deviation of the planning time was reduced by 60% percent when the semantically informed (compressed) graph was employed for semantic planning.

V. CONCLUSIONS

In this paper, we considered the development of a joint semantic mapping and compression framework that simultaneously builds and compresses 3-D probabilistic semantic octree representations. Our framework consists of two parts: a Bayesian multi-class tree-building framework, and an information-theoretic tree-compression scheme. The developed framework is fully probabilistic and allows multi-

resolution abstractions to be tailored to task-relevant and task-irrelevant semantic classes and information. To demonstrate our approach, we compress large semantic maps built from real-world sensor data, and show how the abstractions can be used to improve the performance of planning algorithms over colored (semantic) graphs.

APPENDIX

Proof: [Proposition 3.1] To prove the proposition, we show that $G(n; \beta, \gamma, \alpha) = p(n)G_\Pi(n; \beta, \gamma, \alpha)$ for all $n \in \mathcal{N}(\mathcal{T}_\mathcal{W})$. There are two cases to consider for any $n \in \mathcal{N}_{\text{int}}(\mathcal{T}_\mathcal{W})$: $p(n) = 0$ and when $p(n) > 0$.

The first of these cases is straightforward, since by definition $G(n; \beta, \gamma, \alpha) = 0$ and $G_\Pi(n; \beta, \gamma, \alpha) = 0$. Thus, $G(n; \beta, \gamma, \alpha) = p(n)G_\Pi(n; \beta, \gamma, \alpha) = 0$ when $p(n) = 0$.

We now show $G(n; \beta, \gamma, \alpha) = p(n)G_\Pi(n; \beta, \gamma, \alpha)$ for any $n \in \mathcal{N}(\mathcal{T}_\mathcal{W})$ for which $p(n) > 0$. The proof is given by induction. Consider any $n \in \mathcal{N}(\mathcal{T}_\mathcal{W})$ that is a parent of a leaf, then

$$G(n; \beta, \gamma, \alpha)$$
$$= p(n) \max\{\sum_i \beta_i \text{JS}_\Pi^{Y_i}(n) - \sum_j \gamma_j \text{JS}_\Pi^{Z_j}(n) - \alpha H(\Pi), \ 0\},$$
$$= p(n)G_\Pi(n; \beta, \gamma, \alpha),$$

which follows from the properties of maximum since $p(n) \geq 0$. Now consider any $n \in \mathcal{N}_k(\mathcal{T}_\mathcal{W})$ (i.e., any node at depth k), $k \geq 1$, and assume the hypothesis holds for all $n' \in \mathcal{N}_{k+1}(\mathcal{T}_\mathcal{W})$ for which $p(n') > 0$. Then,

$$G(n; \beta, \gamma, \alpha)$$
$$= p(n) \max\{\sum_i \beta_i \text{JS}_\Pi^{Y_i}(n) - \sum_j \gamma_j \text{JS}_\Pi^{Z_j}(n) - \alpha H(\Pi)$$
$$+ \frac{1}{p(n)} \sum_{n' \in \mathcal{S}} G(n'; \beta, \gamma, \alpha), \ 0\},$$

where $\mathcal{S} = \{n' \in \mathcal{C}(n) : p(n') > 0\}$, $\mathcal{S} \subseteq \mathcal{N}_{k+1}(\mathcal{T}_\mathcal{W})$. The quantity within the max operator can be written as

$$\max\{\sum_i \beta_i \text{JS}_\Pi^{Y_i}(n) - \sum_j \gamma_j \text{JS}_\Pi^{Z_j}(n) - \alpha H(\Pi)$$
$$+ \frac{1}{p(n)} \sum_{n' \in \mathcal{S}} G(n'; \beta, \gamma, \alpha), \ 0\},$$
$$= \max\{\sum_i \beta_i \text{JS}_\Pi^{Y_i}(n) - \sum_j \gamma_j \text{JS}_\Pi^{Z_j}(n) - \alpha H(\Pi)$$
$$+ \frac{1}{p(n)} \sum_{n' \in \mathcal{C}(n)} p(n')G_\Pi(n'; \beta, \gamma, \alpha), \ 0\},$$

where the equality holds from the induction hypothesis and since, for $n' \in \{\bar{n} \in \mathcal{C}(n) : \bar{n} \notin \mathcal{S}\}$, we have $p(n')G_\Pi(n'; \beta, \gamma, \alpha) = 0$ by definition, leading to $G(n; \beta, \gamma, \alpha) = p(n)G_\Pi(n; \beta, \gamma, \alpha)$.

To show the proposition, we prove if $G(n; \beta, \gamma, \alpha) > 0$ then $G_\Pi(n; \beta, \gamma, \alpha) > 0$ and its converse. Pick any $n \in \mathcal{N}(\mathcal{T}_\mathcal{W})$ and assume $G(n; \beta, \gamma, \alpha) > 0$. Then $p(n) > 0$, and so $G(n; \beta, \gamma, \alpha) = p(n)G_\Pi(n; \beta, \gamma, \alpha)$ implying $G_\Pi(n; \beta, \gamma, \alpha) > 0$. Repeating the steps for $G_\Pi(n; \beta, \gamma, \alpha)$, we obtain the result. ∎

REFERENCES

[1] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, June 1989.

[2] S. Thrun, "Learning occupancy grid maps with forward sensor models," *Autonomous Robots*, vol. 15, no. 2, pp. 111–127, September 2003.

[3] Y. Tian, K. Wang, R. Li, and L. Zhao, "A fast incremental map segmentation algorithm based on spectral clustering and quadtree," *Advances in Mechanical Engineering*, vol. 10, no. 2, February 2018.

[4] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Ocotomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous Robots*, vol. 34, pp. 189–206, 2013.

[5] J. Ortiz, A. Clegg, J. Dong, E. Sucar, D. Novotny, M. Zollhöfer, and M. Mukadam, "iSDF: Real-Time Neural Signed Distance Fields for Robot Perception," in *Robotics: Science and Systems*, New York City, NY, USA, June 27-July 1, 2022.

[6] K. Saulnier, N. Atanasov, G. J. Pappas, and V. Kumar, "Information theoretic active exploration in signed distance fields," in *IEEE International Conference on Robotics and Automation (ICRA)*, Paris, France, May 31-August 31, 2020, pp. 4080–4085.

[7] A. Hong, O. Igharoro, Y. Liu, F. Niroui, G. Nejat, and B. Benhabib, "Investigating human-robot teams for learning-based semi-autonomous control in urban search and rescue environments," *Journal of Intelligent & Robotic Systems*, vol. 94, pp. 669–686, June 2019.

[8] N. T. Nguyen, L. Schilling, M. S. Angern, H. Hamann, F. Ernst, and G. Schildbach, "B-spline path planner for safe navigation of mobile robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Prague, Czech Republic, September 27-October 1, 2021, pp. 339–345.

[9] P. Fankhauser, M. Bloesch, and M. Hutter, "Probabilistic terrain mapping for mobile robots with uncertain localization," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3019–3026, October 2018.

[10] E. Zobeidi, A. Koppel, and N. Atanasov, "Dense incremental metric-semantic mapping via sparse gaussian process regression," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Las Vegas, NV, USA, October 25-29, 2020, pp. 6180–6187.

[11] A. Asgharivaskasi and N. Atanasov, "Semantic OcTree Mapping and Shannon Mutual Information Computation for Robot Exploration," *arXiv preprint: 2112.04063*, 2021.

[12] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an open-source library for real-time metric-semantic localization and mapping," in *IEEE International Conference on Robotics and Automation (ICRA)*, Paris, France, May 31-August 31, 2020, pp. 1689–1696.

[13] F. Hauer, A. Kundu, J. M. Rehg, and P. Tsiotras, "Multi-scale perception and path planning on probabilistic obstacle maps," in *IEEE International Conference on Robotics and Automation*, Seattle, WA, USA, May 26-30, 2015, pp. 4210–4215.

[14] R. V. Cowlagi and P. Tsiotras, "Multi-resolution path planning: Theoretical analysis, efficient implementation, and extensions to dynamic environments," in *IEEE Conference on Decision and Control*, Atlanta, GA, USA, December 15-17, 2010, pp. 1384–1390.

[15] D. T. Larsson, D. Maity, and P. Tsiotras, "Information-theoretic abstractions for planning in agents with computational constraints," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7651–7658, October 2021.

[16] S. Kambhampati and L. S. Davis, "Multiresolution path planning for mobile robots," *IEEE Journal of Robotics and Automation*, vol. RA-2, no. 3, pp. 135–145, September 1986.

[17] J. Lim and P. Tsiotras, "Mams-a*: Multi-agent multi-scale a*," in *IEEE International Conference on Robotics and Automation (ICRA)*, Paris, France, May 31-August 31, 2020, pp. 5583–5589.

[18] C. Boutilier and R. Dearden, "Using abstractions for decision-theoretic planning with time constraints," in *AAAI National Conference on Artificial Intelligence*, Seattle, WA, USA, July 31-August 4, 1994, pp. 1016–1022.

[19] G. K. Kraetzschmar, G. P. Gassull, and K. Uhl, "Probabilistic quadtrees for variable-resolution mapping of large environments," *IFAC Proceedings Volumes*, vol. 37, no. 8, pp. 675–680, July 2004.

[20] E. Einhorn, C. Schröter, and H.-M. Gross, "Finding the adequate resolution for grid mapping - cell sizes locally adapting on-the-fly," in *IEEE Conference on Robotics and Automation*, Shanghai, China, May 9-13, 2011, pp. 1843–1848.

[21] E. Nelson, M. Corah, and N. Michael, "Environment model adaptation for mobile robot exploration," *Autonomous Robots*, vol. 42, pp. 257–272, February 2018.

[22] J. Zucker, "A grounded theory of abstraction in artificial intelligence," *Philosophical Transactions of the Royal Society of London, Series B: Biological Sciences 358*, no. 1435, pp. 1293–1309, July 2003.

[23] R. A. Brooks, "Intelligence without representation," *Artificial intelligence*, vol. 47, no. 1-3, pp. 139–159, January 1991.

[24] E. D. Sacerdoti, "Planning in a hierarchy of abstraction spaces," *Artificial Intelligence*, vol. 5, no. 2, pp. 115–135, June 1974.

[25] F. Giunchiglia and T. Walsh, "A theory of abstraction," *Artificial Intelligence*, vol. 57, no. 2-3, pp. 323–389, October 1992.

[26] R. C. Holte and B. Y. Choueiry, "Abstraction and reformulation in artificial intelligence," *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, vol. 358, no. 1435, pp. 1197–1204, July 2003.

[27] M. Ponsen, M. E. Taylor, and K. Tuyls, "Abstraction and generalization in reinforcement learning: A summary and framework," in *Adaptive and Learning Agents*. Springer Berlin Heidelberg, 2010, pp. 1–32.

[28] T. Genewein, F. Leibfried, J. Grau-Moya, and D. A. Braun, "Bounded rationality, abstraction, and hierarchical decision-making: An information-theoretic optimality principle," *Frontiers in Robotics and AI*, vol. 2, no. 27, November 2015.

[29] N. Tishby and D. Polani, "Information theory of decisions and actions," in *Perception-Action Cycle*. Springer New York, December 2010, pp. 601–636.

[30] V. Pacelli and A. Majumdar, "Task-driven estimation and control via information bottlenecks," in *International Conference on Robotics and Automation (ICRA)*, Montreal, Canada, May 20-24, 2019, pp. 2061–2067.

[31] R. G. Gallager, *Information Theory and Reliable Communication*. Wiley, 1968.

[32] N. Tishby, F. C. Pereira, and W. Bialek, "The information bottleneck method," in *Allerton Conference on Communication, Control and Computing*, Monticello, IL, USA, September 1999, pp. 368–377.

[33] D. T. Larsson, D. Maity, and P. Tsiotras, "Q-tree search: an information-theoretic approach toward hierarchical abstractions for agents with computational limitations," *IEEE Transactions on Robotics*, vol. 36, no. 6, pp. 1669–1685, December 2020.

[34] ——, "Information-theoretic abstractions for resource-constrained agents via mixed-integer linear programming," in *Proceedings of the Workshop on Computation-Aware Algorithmic Design for Cyber-Physical Systems*, Nashville, TN, USA, May 18, 2021, pp. 1–6.

[35] ——, "A generalized information-theoretic framework for the emergence of hierarchical abstractions in resource-limited systems," *Entropy*, vol. 24, no. 6, June 2022.

[36] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2nd ed. John Wiley & Sons, 2006.

[37] J. Lin, "Divergence measures based on the Shannon entropy," *IEEE Transactions on Information Theory*, vol. 37, no. 1, pp. 145–151, January 1991.

[38] J. A. Bondy and U. S. R. Murty, *Graph Theory with Applications*. Macmillan Education UK, 1976.

[39] "Pingolh/fchardnet: Fully convolutional hardnet for segmentation in pytorch." [Online]. Available: https://github.com/PingoLH/FCHarDNet

[40] M. Wigness, S. Eum, J. G. Rogers, D. Han, and H. Kwon, "A rugd dataset for autonomous navigation and visual perception in unstructured outdoor environments," in *International Conference on Intelligent Robots and Systems (IROS)*, Macau, China, November 4-8, 2019, pp. 5000–5007.

[41] J. H. Halton, "Algorithm 247: Radical-inverse quasi-random point sequence," *Communications of the ACM*, vol. 7, no. 12, pp. 701–702, December 1964.

[42] J. Lim and P. Tsiotras, "A generalized A* algorithm for finding globally optimal paths in weighted colored graphs," in *IEEE International Conference on Robotics and Automation (ICRA)*, Xi'an, China, May 30-June 5 2021, pp. 7503–7509.

[43] J. A. Reeds and L. A. Shepp, "Optimal paths for a car that goes both forwards and backwards," *Pacific Journal of Mathematics*, vol. 145, no. 2, pp. 367–393, October 1990.