2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Communication-Aware Map Compression for Online Path-Planning

Evangelos Psomiadis, Dipankar Maity, Panagiotis Tsiotras

Abstract— **This paper addresses the problem of the communication of optimally compressed information for mobile robot path-planning. In this context, mobile robots compress their current local maps to assist another robot in reaching a target in an unknown environment. We propose a framework that sequentially selects the optimal level of compression, guided by the robot's path, by balancing map resolution and communication cost. Our approach is tractable in close-to-real scenarios and does not necessitate prior environment knowledge. We design a novel decoder that leverages compressed information to estimate the unknown environment via convex optimization with linear constraints and an encoder that utilizes the decoder to select the optimal compression. Numerical simulations are conducted both in a large close-to-real map and a maze map and compared with two alternative approaches. The results confirm the effectiveness of our framework in assisting the robot reach its target by reducing transmitted information, on average, by approximately 50%, while maintaining satisfactory performance.**

I. Introduction

Advancements in the field of multi-robot decision-making enable teams of robots to carry out complex tasks such as search and rescue operations [1], autonomous delivery [2], or even space missions [3]. These operations often center around collaborative navigation in unknown environments, where the robots engage in continuous information exchange. However, to fully harness the potential of their communication network and optimize performance, the robots must be aware of their bandwidth limitations, and incorporate those in their control and decision-making process. The problem of multi-robot path-planning under bandwidth constraints is an active area of research and several approaches have been proposed, handling different aspects of the problem [4].

Prior robot control algorithms have treated communication as an afterthought. For example, in [5], given a data set, the authors assign metrics to assess the significance of data points and decide their communication for exploration missions. In a recent work for navigation/path-planning [6], the authors compress a 3D Scene Graph, given a high-resolution path by keeping a specific number of nodes. Our approach compresses the essential information for path-planning online, integrating the compressed map into the planning loop.

The work was supported by the ARL grant DCIST CRA W911NF-17-2-0181.

E. Psomiadis and P. Tsiotras are with the D. Guggenheim School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA, 30332-0150, USA. Email: {epsomiadis3, tsiotras}@gatech.edu

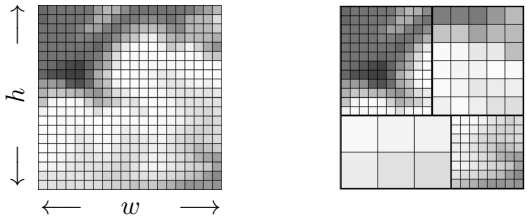
D. Maity is with the Department of Electrical and Computer Engineering, University of North Carolina at Charlotte, NC, 28223-0001, USA. Email: dmaity@charlotte.edu

The coupled problem between compression (quantization) and control has been an active area of research for several decades in the controls community [7]–[10]. The choice of the optimal quantizer even for a given control objective (e.g., a quadratic cost function) is, however, an intractable problem [11], [12]. Instead of designing the optimal quantizer, an alternative approach is to select the best quantizer at each time instance from a given set of quantizers [13], [14]. This results in a tractable linear program (LP). We adopt this approach, where a set of quantizers is available to the robots to compress their perception data before transmitting it to another robot in their team.

Related Work: Our primary focus is determining what information to communicate in the context of a multi-agent navigation problem involving agents with different objectives. In [15], the authors introduce ConTaCT, a policy that addresses when to communicate information in multi-agent navigation scenarios by solving a decentralized Markov Decision Process with the team reward dependent on the joint action space. In [16] the agents communicate whenever there is an inconsistency in their shared belief. In [17] the authors address the problem of deciding what information to communicate using OCBC, an algorithm that employs forward simulations and a bandit-based combinatorial optimization to evaluate observations. This approach can become computationally intractable when increasing the robots' field of view, as it increases the number of candidate observations to assess. Additionally, learning-based methods have also been employed. In [18], the authors propose an architecture comprising a Convolutional Neural Network and a Graph Neural Network that compress and communicate information among robots for decentralized sequential path-planning.

Contributions: In this paper, we assume a team of mobile robots that autonomously choose the optimal way to compress map data to assist another robot navigate an unknown environment. Our approach does not require prior environment knowledge, like learning-based methods, and is tractable in large maps, regardless of the robot configurations.

We propose a novel decoder-encoder pair architecture to estimate the unknown environmental occupancy values and select the optimal compression, utilizing a given set of quantizers. We validate the effectiveness of our framework through simulations conducted on both a large, real-world-like map and a maze map. While our simulations involve a single pair of robots, our framework can be readily extended to multiple robots.



(a) (b)

Fig. 1: (a) Finest resolution occupancy grid; (b) Compressed (i.e., quantized) occupancy grid.

II. Preliminaries: Grid World and Abstractions

We assume that the environment is represented by an occupancy grid M in 2D[1], with N being the total number of cells in M. A robot with onboard sensors can observe a 2D occupancy grid of size $w \times h \leq N$ as shown in Figure 1(a). The occupancy values of the grid cells are stored in the vector $x \in \mathbb{R}^N$. The j-th component of x, denoted by $[x]_j$, is in the range $[0, 1]$ for all $j = 1, \ldots, N$. Here $[x]_j$ denotes the traversability of the cell, where $[x]_j = 1$ indicates a non-traversable cell and $[x]_j = 0$ denotes an obstacle-free cell. A compressed representation of the occupancy grid is described by an *abstraction*, as in Figure 1(b).

Each abstraction is associated with a *compression template* that generates a compressed representation of the occupancy grid. This can be considered as a guide, indicating what and how the grid cells are going to be abstracted. The abstracted representation is a multi-resolution occupancy grid, where the occupancy of a compressed cell is determined by the occupancy values of the finest resolution cells that make up the compressed cell. The occupancy value of a compressed cell can be determined using existing techniques, such as wavelets [19], k-class trees [20] or information bottleneck methods [21]. For simplicity, in this study, we adopt the approach of computing the occupancy value of the compressed cell as the average of the underlying occupancy values of the finest resolution cells. This approach aligns with the principles of the information bottleneck method [21]. Therefore, each abstraction can be thought of as a linear mapping $\mathcal{A}^\theta : [0, 1]^N \to [0, 1]^{k_\theta}$, where $k_\theta \leq wh \leq N$ is the number of cells in the compressed occupancy grid employing abstraction θ. That is, for a given high resolution occupancy map $x \in [0, 1]^N$, the occupancy values of the cells in abstraction θ will be $o = \mathcal{A}^\theta x$, where $\mathcal{A}^\theta \in \mathbb{R}^{k_\theta \times N}$. Since we assume that the occupancy value of a compressed cell is the average of the occupancy values of the underlying finest resolution cells, the matrix $\mathcal{A}^\theta$ is row stochastic for all $\theta \in \Theta$, where $\Theta = \{1, \ldots, K\}$ is the set of available abstractions.

A. Communication of Abstracted Environments

Let a pair of robots, robot A and robot B, that have agreed on an abstraction set Θ. At every timestep t, robot B selects an abstraction θ to compress its observed occupancy grid and transmits it to robot A. Specifically, robot B transmits the pair (o_t, θ), where $o_t = \mathcal{A}^\theta_t x$, with x containing the finest resolution measurements sensed by robot B. Robot A knows $\mathcal{A}^\theta_t$ and attempts to reconstruct x from o_t.

Let n_m and n_i denote the number of bits required to transmit an occupancy value $[o]_j$ and an abstraction index θ, respectively. Therefore, for abstraction θ, the total number of bits n_θ required to send the abstracted grid is given by:

$$n_\theta = k_\theta n_m + n_i. \tag{1}$$

If robot B were to send the finest resolution occupancy grid at each timestep, the required bits would be equal to whn_m.

[1]The framework extends in a straightforward manner to 3D environments and more complicated discretizations.

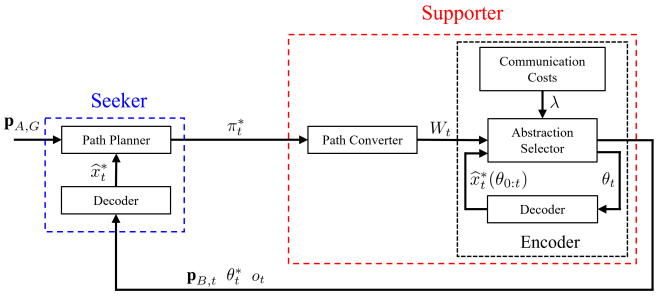


Fig. 2: Flowchart that presents the proposed framework's architecture at timestep t. The Seeker shares its current optimal path π^*_t to the Supporter. The Supporter utilizes this to select the optimal abstraction of its local map $\theta^*_t \in \Theta$ and sends it to the Seeker to aid in reaching its target.

III. Problem Formulation

Consider a pair of mobile robots, a Seeker and a Supporter, that navigates through an unfamiliar environment $M \subset \mathbb{R}^2$ repleted with static obstacles. Let $\mathbf{p}_{A,t}$, $\mathbf{p}_{B,t} \in M$ be the Seeker and Supporter's positions respectively at timestep t, and let $u_t \in U$ denote the Seeker's control action at time t selected from a finite set of control actions U. The robots are equipped with sensors capable of observing a portion of the environment (local map) as they traverse it. The Seeker's objective is to reach a designated target in minimum time by following a path generated by an online path-planning algorithm. In contrast, the Supporter follows a predefined path and aims to assist the Seeker in achieving its objective by transmitting informative abstractions of its local map to the Seeker at each timestep. In this work, we do not adhere to a specific way to design the Supporter's path, but we consider it to be given and prove the effectiveness of our algorithm regardless of it. The Supporter's role can be likened to that of a drone assigned to reach a separate target or even a satellite in orbit, passing over the environment of the Seeker. Its path is determined a priori by a different objective and cannot be altered. The proposed framework is shown in Figure 2, and the roles of its components are delineated in Section IV.

A. Problem Statement

Considering as inputs the initial positions of the Seeker and the Supporter, $\mathbf{p}_{A,0}$ and $\mathbf{p}_{B,0}$, along with the Supporter's

predefined path and the Seeker's target $\mathbf{p}_{A,G}$ in a global reference frame, we design a framework to online select the optimal abstraction θ^*, from a given set of abstractions Θ, to compress the Supporter's local map. The Supporter's encoder selects the abstractions, driven by the Seeker's transmitted path at every timestep. Meanwhile, the Seeker utilizes the accumulated measurements $o_{0:t} = \{o_0, o_1, \ldots, o_t\}$ up to time t to compute the control action u_t to reach its ultimate destination in minimum time (e.g., shortest path).

IV. Framework Architecture

A. Decoder

The decoder's primary role (see Figure 2) is to provide estimates for the vector $x \in [0,1]^N$ containing the occupancy values of the cells of M, where N is the total number of cells in M. To achieve this, the decoder leverages both the past and present Seeker's measurements and the Supporter's choices for abstractions. Recall that the Supporter's abstraction at timestep t is described by $\mathcal{A}_t^\theta x = o_t$, where $\mathcal{A}_t^\theta \in \mathbb{R}^{k_\theta \times N}$, while a Seeker's measurement of the j-th cell can be described by $\alpha_t^j x = o_t^j$, where $\alpha_t^j \in \mathbb{R}^N$ has zeros everywhere except at the j-th element. Hence, the Seeker's measurements as well as the Supporter's abstractions can be described by a set of linear equality and inequality constraints:

$$C_t = \left\{ x \in \mathbb{R}^N : \begin{array}{c} \mathcal{A}_{0:t} x = o_{0:t}, \\ 0 \le [x]_j \le 1,\ j = 1, \ldots, N \end{array} \right\}, \quad (2)$$

where $\mathcal{A}_{0:t} = \{\mathcal{A}_0, \mathcal{A}_1, \ldots, \mathcal{A}_t\}$ and $\mathcal{A}_t = \{\mathcal{A}_t^\theta, \alpha_t\}$. Let k_t be the number of equality constraints in (2), equal to the number of past and current Seeker's measurements and Supporter's abstractions, with redundant, linearly dependent equations removed. It becomes evident that at each timestep, the rows of $\mathcal{A}_{0:t} \in \mathbb{R}^{k_t \times N}$ and $o_{0:t} \in \mathbb{R}^{k_t}$ might increase as new measurements and abstractions are added.

In case the true values of certain elements of x are not perfectly known, it is imperative to establish a systematic method for computing estimates, utilizing the set C_t. To provide such estimates, we assume that the occupancy vector x is a multivariate random variable following a distribution, known to both the Seeker and the Supporter. Leveraging principles of stochastic estimation [22], we find the conditional expectation for each element of x within C_t. This is achieved by identifying the point in C_t that minimizes the variance of the estimation error. Therefore, the vector $\widehat{x}_t^*$ containing the estimates of x is given by:

$$\widehat{x}_t^* = \underset{\widehat{x}_t \in C_t}{\arg\min} \sum_{j=0}^{N} \mathbb{E}[[x]_j - [\widehat{x}_t]_j]^2. \quad (3)$$

Given a distribution for x, (3) can be transformed into a convex optimization problem with linear constraints.

Proposition 1: Let x follow a distribution with mean μ. Then,

$$\widehat{x}_t^* = \underset{\widehat{x}_t \in C_t}{\arg\min} \|\widehat{x}_t - \mu\|^2. \quad (4)$$

Proof: Equation (3) can be equivalently written as:

$$\widehat{x}_t^* = \underset{\widehat{x}_t \in C_t}{\arg\min} \mathbb{E}[\|x - \widehat{x}_t\|^2]. \quad (5)$$

Notice that,

$$\begin{aligned} \mathbb{E}[\|x - \widehat{x}_t\|^2] &= \mathbb{E}[\|x\|^2] + \|\widehat{x}_t\|^2 - 2\mathbb{E}[x]^\top \widehat{x}_t \\ &= \mathrm{tr}(\Sigma) + \|\mu\|^2 + \|\widehat{x}_t\|^2 - 2\mu^\top \widehat{x}_t \\ &= \mathrm{tr}(\Sigma) + \|\widehat{x}_t - \mu\|^2, \end{aligned}$$

where $\Sigma = \mathbb{E}[xx^\top] - \mathbb{E}[x]\mathbb{E}[x]^\top$ is the covariance matrix of x and $\mathrm{tr}(\cdot)$ denotes the trace of a matrix. Thus,

$$\widehat{x}_t^* = \underset{\widehat{x}_t \in C_t}{\arg\min} \mathbb{E}[\|x - \widehat{x}_t\|^2] = \underset{\widehat{x}_t \in C_t}{\arg\min} \|\widehat{x}_t - \mu\|^2.$$

■

Given that C_t is a convex set (polyhedron) and the objective function is quadratic, the optimization problem is convex. It is further noteworthy that the optimal solution $\widehat{x}_t^*$ depends only on the mean of the distribution. In this work, we set $\mathbb{E}[[x]_j] = 0.5$, since the decoder lacks any prior information about the environment and $[x]_j$ is bounded between 0 and 1. Therefore:

$$\widehat{x}_t^* = \underset{\widehat{x}_t \in C_t}{\arg\min} \|\widehat{x}_t - \tfrac{1}{2}\mathbf{1}\|^2, \quad (6)$$

where $\mathbf{1}$ is a vector of all ones.

B. Path Planner

Let $G = (V, E)$ represent the graph associated with the occupancy grid environment M, where V denotes the set of vertices and E the set of edges. Each vertex in V corresponds to a specific cell in M (with a slight abuse of notation, we will use $\mathbf{p}$ to denote both cell positions and graph vertices). Two vertices are deemed connected if the Seeker can move between the two utilizing a control action $u \in U$. We assume that the estimated time to traverse a cell is proportional to its occupancy value (difficulty to traverse) plus a constant (movement penalty). The estimated cost of traversing a vertex is therefore given by [23]:

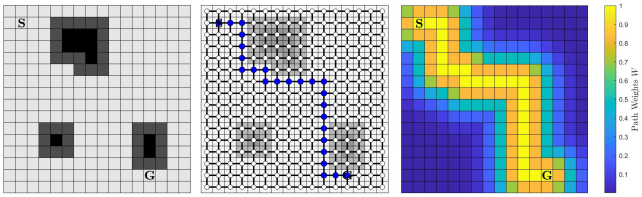
$$c_\epsilon(\mathbf{p}) = \begin{cases} \widehat{x}_\mathbf{p} + a, & \text{if } \mathbf{p} \in P_\epsilon, \\ N(\epsilon + a), & \text{otherwise}, \end{cases} \quad (7)$$

where $\widehat{x}_\mathbf{p} \in [0,1]$ is the estimated occupancy value of the cell at position $\mathbf{p}$, a is a constant cost for traversing a cell, N is the total number of vertices or cells in M, and $P_\epsilon = \{\mathbf{p} \in V : \widehat{x}_\mathbf{p} \le \epsilon\}$ designates the set of cells meeting a feasibility condition, where $\epsilon \in [0,1]$ is a scalar.

Let Π denote the set of paths with the first element being the Seeker's current position $\mathbf{p}_{A,t}$ and the last element being its goal location $\mathbf{p}_{A,G}$. Then, the optimal path is given by:

$$\pi^* = \underset{\pi \in \Pi}{\arg\min} \sum_{\mathbf{p} \in \pi} c_\epsilon(\mathbf{p}). \quad (8)$$

When $\pi \subseteq P_\epsilon$, the path is referred to as an ϵ-feasible path [23]. By setting the second part of (7) larger than the cost of any feasible path, we exclude infeasible vertices, unless no feasible path is available. This ensures that the path-planning algorithm will always find a path.



(a) (b) (c)

Fig. 3: (a) Example of a discretized environment. **S** denotes the starting cell while **G** denotes the target; (b) Associated graph of the discretized environment employing the set of actions $U = \{\text{UP, DOWN, LEFT, RIGHT}\}$, along with the optimal path (blue nodes) with cost function (7); (c) Path weights computed using (9).

Figure 3(a) presents a simple example of an occupancy grid environment with obstacles, while Figure 3(b) illustrates the associated graph and the optimal path (computed, for instance, using Dijkstra's graph search algorithm [24], with cost function (7)).

C. Path Converter

The Seeker sends its current optimal path π_t^* obtained from (8) to the Supporter. The Supporter utilizes this information to guide its abstraction selection process. This is achieved by assigning weights to each cell within M based on its proximity to the path. To compute these weights, we employ a normalized Gaussian function:

$$w_t(\mathbf{p}) = \max_{\mathbf{p}_\pi \in \pi_t^*} e^{-\frac{\|\mathbf{p}-\mathbf{p}_\pi\|^2}{2\sigma^2}}, \qquad \mathbf{p} \in M, \tag{9}$$

where σ is a scalar characterizing the width around the path.

Figure 3(c) illustrates the path weights after applying (9) to the example in Figure 3(b).

D. Encoder

The encoder's role is to select the optimal abstraction from a given set, to transmit to the Seeker. This selection is conducted with a focus on both navigation and communication aspects, considering the path weights $w_t(\mathbf{p})$ and penalizing abstractions based on their required transmission bandwidth.

Let $\theta \in \Theta$ denote an abstraction, where Θ is defined in Section II-A. The optimal abstraction at each timestep t is derived through the minimization of the following criterion:

$$J_t(\theta_{0:t}) = \|W_t \circ (\widetilde{x}_t - \widehat{x}_t^*(\theta_{0:t}))\|^2 + \lambda(\theta_t), \tag{10a}$$

$$\theta_t^* = \arg\min_{\theta_t \in \Theta} J_t(\theta_{0:t}), \tag{10b}$$

where W_t is the vector containing the weights $w_t(\mathbf{p})$ of every cell, $\widetilde{x}_t \in \mathbb{R}^N$ is the total sensed occupancy grid by the Supporter until timestep t, $\widehat{x}_t(\theta_{0:t}) \in \mathbb{R}^N$ is the estimation vector (depending on the history of abstractions), $\circ$ is the Hadamard product, and $\lambda(\theta_t)$ is the communication/bandwidth cost assigned to each abstraction. Similar to previous works in classical control [14], we do not impose any specific structure on $\lambda(\theta_t)$, while considering that the cost is proportional to the number of compressed cells in an abstraction. This approach enables us to capture the idea of penalizing abstractions based on their resolution.

Algorithm 1 The Encoder's Algorithm

Input: W_t, $\mathcal{A}_{0:t-1}$, $o_{0:t-1}$, $\mathbf{p}_{A,0:t}$, $\mathbf{p}_{B,0:t}$
Output: θ_t^*, o_t

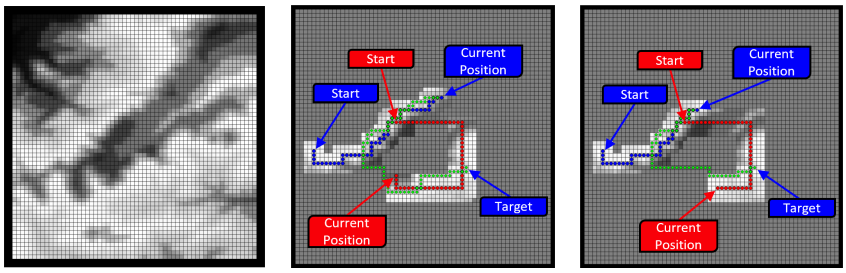
1: **for all** $\mathbf{p} \in (L_{B,t} \cap M_{A,t}) \cup (L_{A,t} \cap M_{B,t})$ **do**
2: $\quad(\mathcal{A}, o) \leftarrow \textsc{Update}(\mathcal{A}_{0:t-1}, o_{0:t-1})$
3: **end for**
4: **for all** $\theta \in \Theta$ **do**
5: $\quad(\mathcal{A}_t^\theta, o_t^\theta) \leftarrow \textsc{Abstractions}(L_{B,t}, \theta)$
6: $\quad(\mathcal{A}_{0:t}, o_{0:t}) \leftarrow \textsc{Independent}\left(\begin{bmatrix} \mathcal{A} & o \\ \mathcal{A}_t^\theta & o_t^\theta \end{bmatrix}\right)$
7: $\quad\widehat{x}_t^* \leftarrow \textsc{Decoder}(\mathcal{A}_{0:t}, o_{0:t})$ using (6)
8: $\quad J_t(\theta) \leftarrow \textsc{ComJ}(W_t, \widehat{x}_t^*, \widetilde{x}_t(\mathbf{p}_{B,0:t}), \lambda)$ using (10a)
9: **end for**
10: $\theta_t^* \leftarrow \arg\min J_t$
11: $o_t \leftarrow o_t^{\theta_t^*}$
12: **return** θ_t^*, o_t

It is important that the Supporter's encoder selects the optimal abstraction for the specific Seeker's decoder (see Figure 2). Therefore, the estimation vector $\widehat{x}_t(\theta_{0:t})$ is computed by solving (6) with the constraint set given by:

$$C_t^\theta = \left\{ x \in \mathbb{R}^N : \begin{bmatrix} \mathcal{A} \\ \mathcal{A}_t^\theta \end{bmatrix} x = \begin{bmatrix} o \\ o_t^\theta \end{bmatrix} \right. \\ \left. 0 \le [x]_j \le 1,\ j = 1, \dots, N \right\}, \tag{11}$$

where $\mathcal{A}_t^\theta$ and o_t^θ denote the matrix of the candidate abstraction θ and the occupancy values, respectively, at timestep t (see Section II) and $(\mathcal{A}, o)$ is described in the next paragraph.

The encoder's algorithm is given in Algorithm 1. Here $L_{A,t} = L_{A,t}(\mathbf{p}_{A,t}) \subseteq M$ and $L_{B,t} = L_{B,t}(\mathbf{p}_{B,t}) \subseteq M$ are sets that contain the cells of the Seeker and Supporter's current local map respectively, and $M_{A,t} = M_{A,t}(\mathbf{p}_{A,0:t}) \subseteq M$ and $M_{B,t}(\mathbf{p}_{B,0:t}) \subseteq M$ are sets that contain all the sensed/finest resolution cells from timestep 0 to t of the Seeker and the Supporter respectively. Lines 1-3 in Algorithm 1 apply the function UPDATE to incorporate the values of the finest resolution cells of the Supporter's local map that were just measured by the Seeker directly to the previous measurements, constructing $(\mathcal{A}, o)$. These cells are excluded from the abstraction process. Moreover, they inform the Supporter if the Seeker has just measured cells previously included in transmitted abstractions, and add them to $(\mathcal{A}, o)$. Line 5 uses the function ABSTRACTIONS to compute the pair $(\mathcal{A}_t^\theta, o_t^\theta)$ by applying abstraction θ to $L_{B,t}$, while Line 6 uses the function INDEPENDENT to concatenate all the measurements and uses a method (i.e., Gaussian elimination) to exclude linear dependent equations. Lines 4-9 perform an exhaustive search to find the optimal abstraction, as the available set of abstractions is limited. Note that the simulation time does not depend heavily on the environment's size, but it rather depends on the abstraction set, particularly on the number of finest resolution cells in a compressed cell.



(a) Realistic Map (b) AS framework ($t = 51$) (c) FI framework ($t = 40$)

(d) U framework (final) (e) AS framework (final) (f) FI framework (final)

Fig. 4: Simulation example using the realistic map. Figures (b)-(f) illustrate the Seeker's estimated map at different instances. The Supporter's path is illustrated with red and its initial position is $\mathbf{p}_{B,0} = (26, 36)$. The cells that the Seeker has already traversed are presented with blue, while green is the current path constructed by its path-planning algorithm.

V. Experiments

In this section, we present the simulation results to validate the effectiveness of our framework. We conducted 500 simulations on each of two different 2D maps: a realistic map (64×64) with probabilistic occupancy values (Figure 4(a)), and a maze (30×30) with deterministic values (Figure 5). A single pair of a Seeker and a Supporter is employed, with both robots initiating movement simultaneously and traversing one cell per timestep. The Supporter has the ability to move over obstacles (i.e., it is an aerial vehicle). The Seeker's local map $L_{A,t}$ size is 5×5 cells in the first scenario and 3×3 cells in the second. The Supporter has a field of view $L_{B,t}$ of 7×7 cells in both scenarios. Both robots are positioned at the center of their respective local maps and the Seeker's control set is U ={UP, DOWN, LEFT, RIGHT}. While the framework can adeptly handle more complex simulation environments (e.g., different discretization, robots' field of view, control sets) the above assumptions were chosen for their simplicity and ease of interpretation regarding the framework's effectiveness. To this end, it is assumed that a lower-level controller manages the robots' dynamics, while our framework serves as a higher-level controller.

In the first scenario, we tested different Supporter's predefined paths while maintaining the same initial and final positions of the Seeker. In the second scenario, we vary the initial and final positions of the Seeker while retaining the same predefined path. The Seeker's Path Planner uses the cell cost given in (7) with $a = 0.025$ and $\epsilon = 0.501$. The Supporter's encoder utilizes a finite set of 10 abstractions as shown in Figure 6 and with parameters in (9) $\sigma = 10$ and $\sigma = 3.33$ for the real-world-like and maze map, respectively.

A. Performance Metrics

We compare our (Abstraction Selection - AS) framework with two alternatives: a Fully-Informed (FI) framework and an Uninformed (U) framework. In the FI framework, the Supporter transmits all the new measurements contained in $L_{B,t}$ at each timestep t. In the U framework, the Seeker reaches its destination without assistance from the Supporter.

As explained in Section IV-B, we assumed that the time required to traverse a cell is proportional to the cell's cost. Hence, the total time taken by the Seeker to reach its target is proportional to the accumulated cost $\mathcal{C}$. Let π_f include the cells that the Seeker traversed until it reached its target, without excluding duplicate cells. Hence, $\mathcal{C} = \sum_{\mathbf{p}\in\pi_f} c_\epsilon(\mathbf{p})$.

We assess our framework's effectiveness in assisting the Seeker to reach its destination, by calculating the total number of simulations in which the Seeker had the highest $\mathcal{C}$, in comparison to the other two frameworks, and we classify these simulations as "failure". Meanwhile, we classify simulations that resulted in the same $\mathcal{C}$ as "neutral".

We also compute the average time ratio:

$$r_{\text{time},i} = \frac{1}{n_{\text{sim}}} \sum_{s=0}^{n_{\text{sim}}} \frac{\mathcal{C}_i(s)}{\mathcal{C}_o(s)}, \tag{12}$$

where n_{sim} is the total simulation number, $\mathcal{C}_o(s)$ is the accumulated cost using the optimal framework at simulation s, i is the framework index (i.e., $i = 1$ for FI, $i = 2$ for AS, and $i = 3$ for U framework), and $\mathcal{C}_i(s)$ is the accumulated cost of framework i at simulation s.

Additionally, we evaluated the performance of our framework in reducing the amount of information sent at each timestep, by calculating the average ratio of bits sent by our

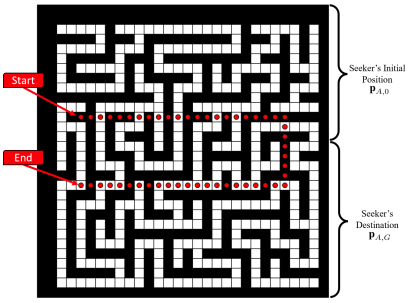


Fig. 5: Maze Map. The Supporter's path is defined with red.

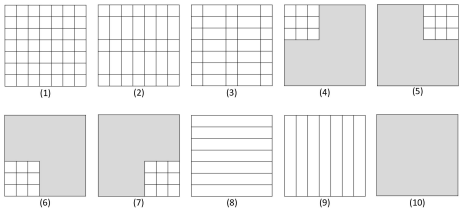


Fig. 6: Set of Abstractions used for the simulations. The grey area denotes no information about these cells

framework and the bits sent by the FI framework as follows:

$$r_{\text{bits}} = \frac{1}{n_{\text{sim}}} \sum_{s=0}^{n_{\text{sim}}} \frac{\sum_{t=0}^{T_{B,\theta}(s)} n_{\theta,t}(s)}{\sum_{t=0}^{T_{B,f}(s)} n_{f,t}(s)}, \tag{13}$$

where $n_{\theta,t}(s)$ is given in (1) and denotes the bits sent by our framework's Supporter using abstraction θ at timestep t and simulation s, $n_{f,t}(s)$ are the bits sent by the FI framework at timestep t and simulation s, and $T_{B,\theta}(s)$ and $T_{B,f}(s)$ is the time horizon that the Supporter transmits information at simulation s for the AS and FI framework, respectively. The parameter values in (1) are $n_m = 12$, $n_i = 4$.

B. Variation of Supporter's Path

In the realistic map environment (Figure 4(a)), we conducted 500 simulations with different predefined paths of the Supporter. The Seeker's initial position is $\mathbf{p}_{A,0} = (6, 29)$ and the destination is $\mathbf{p}_{A,G} = (43, 25)$. The Supporter's initial position on the map is arbitrary and determines which of the four different paths it will follow with $T_{B,\max} = 60$.

Table I presents the results. We conclude that the FI framework produced the highest $\mathcal{C}$ for the smallest number of simulations. However, we also observe that there were 28 simulations where the FI framework had the highest $\mathcal{C}$. This means that information might not always be beneficial for the Seeker but, on the contrary, it might be misleading. Informing the Seeker about obstacle-free areas that lead to potential dead-ends may cause it to enter these areas. Conversely, alerting the Seeker to blocked areas may lead it to mistakenly avoid regions with clear paths nearby.

In conclusion, our framework increased, on average, the cost by 26.9% whereas the FI framework increased it by 2.2%, and the U framework by 141.0%. However, our framework also achieved a 62.7% reduction in transmitted information, while, on average, maintaining a satisfactory performance, compared to the other two alternatives.

TABLE I: Realistic Map Results

Framework	Fully-Informed	Abstraction Selector	Uninformed
failures	28	84	365
neutral	11	11	11
r_{time}	1.022	1.269	2.410
r_{bits}	-	0.373	-

TABLE II: Maze Results

Framework	Fully-Informed	Abstraction Selector	Uninformed
failures	28	59	306
neutral	77	77	77
r_{time}	1.055	1.148	1.343
r_{bits}	-	0.564	-

Figure 4 illustrates one of the 500 conducted simulations on a real-world-like environment. In this example, we observe the effectiveness of our framework (Figure 4(b)) in providing the Seeker with information that prompts it to change direction and follow the correct path faster than the U framework. This results in similar performance to the FI framework, while transmitting fewer bits. Considering the simulation time, on average, the encoder's algorithm took 0.75sec per timestep when running on MATLAB with the MOSEK 10.1 optimizer on a computer with a 2.3 GHz, 8-Core, 11th Gen Intel Core i7-11800H CPU and 16GB RAM.

C. Variation of Seeker's Initial and Final Positions

In the maze map (Figure 5), we run 500 simulations for different initial positions of the Seeker $\mathbf{p}_{A,0}$ and the target $\mathbf{p}_{A,G}$, while keeping the Supporter's path the same. The initial position of the Supporter is $\mathbf{p}_{B,0} = (5, 19)$ and $T_{B,\max} = 49$.

Table II presents the simulation results. Our framework increased, on average, the cost by 14.8% whereas the FI framework increased it by 5.5%, and the U framework by 34.3%. Nonetheless, our framework also managed to reduce transmitted information by 43.6%, while increasing the cost only by 9.3%, compared to the optimal FI framework.

VI. Conclusions

This paper addresses the challenge of determining the optimal information compression for communication in the context of mobile robot path-planning. We assume a team of mobile robots that compress their local maps to assist another robot reach a destination in an unfamiliar environment. In contrast with existing methods, our framework does not require prior knowledge of the environment and is effective for various robot configurations and map sizes. Simulation results validate the effectiveness of our framework. On average, our framework reduced the amount of information by approximately 50% while maintaining satisfactory performance. In the future, we plan to extend our framework to the multi-robot path-planning problem. We also intend to design a more sophisticated search method, utilizing abstraction dependence, to increase the set of abstractions.

REFERENCES

[1] H. Sugiyama, T. Tsujioka, and M. Murata, "Collaborative movement of rescue robots for reliable and effective networking in disaster area," in *International Conference on Collaborative Computing: Networking, Applications and Worksharing*, San Jose, CA, Dec. 19-22, 2005.

[2] O. Salzman and R. Stern, "Research challenges and opportunities in multi-agent path finding and multi-agent pickup and delivery problems," in *19th International Conference on Autonomous Agents and MultiAgent Systems*, Auckland, New Zealand, May 9-13, 2020, pp. 1711–1715.

[3] S. B. Kesner, J.-S. Plante, P. J. Boston, T. Fabian, and S. Dubowsky, "Mobility and power feasibility of a microbot team system for extraterrestrial cave exploration," in *IEEE International Conference on Robotics and Automation (ICRA)*, Rome, Italy, April 10-14, 2007, pp. 4893–4898.

[4] J. Gielis, A. Shankar, and A. Prorok, "A critical review of communications in multi-robot systems," *Current Robotics Reports*, vol. 3, no. 4, pp. 213–225, Aug. 2022.

[5] M. E. Kepler and D. J. Stilwell, "An approach to reduce communication for multi-agent mapping applications," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Las Vegas, NV, Oct. 25-January 24, 2020-2021, pp. 4814–4820.

[6] Y. Chang, L. Ballotta, and L. Carlone, "D-lite: Navigation-oriented compression of 3d scene graphs under communication constraints," 2023, arXiv:2209.06111.

[7] D. F. Delchamps, "Stabilizing a linear system with quantized state feedback," *IEEE Transactions on Automatic Control*, vol. 35, no. 8, pp. 916–924, Aug. 1990.

[8] R. W. Brockett and D. Liberzon, "Quantized feedback stabilization of linear systems," *IEEE Transactions on Automatic Control*, vol. 45, no. 7, pp. 1279–1289, July 2000.

[9] G. N. Nair and R. J. Evans, "Stabilizability of stochastic linear systems with finite feedback data rates," *SIAM Journal on Control and Optimization*, vol. 43, no. 2, pp. 413–436, July 2004.

[10] V. Kostina and B. Hassibi, "Rate-cost tradeoffs in control," *IEEE Transactions on Automatic Control*, vol. 64, no. 11, pp. 4525–4540, Nov. 2019.

[11] M. Fu, "Lack of separation principle for quantized linear quadratic gaussian control," *IEEE Transactions on Automatic Control*, vol. 57, no. 9, pp. 2385–2390, Sept. 2012.

[12] S. Yüksel, "A note on the separation of optimal quantization and control policies in networked control," *SIAM Journal on Control and Optimization*, vol. 57, no. 1, pp. 773–782, 2019.

[13] D. Maity and P. Tsiotras, "Optimal controller synthesis and dynamic quantizer switching for linear-quadratic-Gaussian systems," *IEEE Transactions on Automatic Control*, vol. 67, no. 1, pp. 382–389, Jan. 2022.

[14] ——, "Optimal quantizer scheduling and controller synthesis for partially observable linear systems," *SIAM Journal on Control and Optimization*, vol. 61, no. 4, pp. 2682–2707, Aug. 2023.

[15] V. Unhelkar and J. Shah, "Contact: Deciding to communicate during time-critical collaborative tasks in unknown, deterministic domains," in *AAAI Conference on Artificial Intelligence*, Phoenix, AZ, Feb. 12-17, 2016.

[16] F. Wu, S. Zilberstein, and X. Chen, "Online planning for multi-agent systems with bounded communication," *Artificial Intelligence*, vol. 175, no. 2, pp. 487–511, Feb. 2011.

[17] R. Marcotte, X. Wang, D. Mehta, and E. Olson, "Optimizing multi-robot communication under bandwidth constraints," *Autonomous Robots*, vol. 44, no. 1, pp. 43–55, Jan. 2020.

[18] Q. Li, F. Gama, A. Ribeiro, and A. Prorok, "Graph neural networks for decentralized multi-robot path planning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Macau, China, Nov. 4-8, 2019, pp. 11 785–11 792.

[19] R. V. Cowlagi and P. Tsiotras, "Multiresolution motion planning for autonomous agents via wavelet-based cell decompositions," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 5, pp. 1455–1469, Oct. 2012.

[20] 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.

[21] 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, Dec. 2020.

[22] J. L. Speyer and W. H. Chung, *Stochastic Processes, Estimation, and Control*. Los Angeles, CA: Society for Industrial and Applied Mathematics, 2008.

[23] 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, Oct. 2021.

[24] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, Dec. 1959.