4CNet: A Confidence-Aware, Contrastive, Conditional, Consistency Model for Robot Map Prediction in Multi-Robot Environments

Aaron Hao Tan, *IEEE Student Member*, Siddarth Narasimhan, Goldie Nejat, *IEEE Member*

Abstract— **Mobile robots in unknown cluttered environments with irregularly shaped obstacles often face sensing, energy, and communication challenges which directly affect their ability to explore these environments. In this paper, we introduce a novel deep learning method, Confidence-Aware Contrastive Conditional Consistency Model (*4CNet*), for mobile robot map prediction during resource-limited exploration in multi-robot environments. *4CNet* uniquely incorporates: 1) a conditional consistency model for map prediction in irregularly shaped unknown regions, 2) a contrastive map-trajectory pretraining framework for a trajectory encoder that extracts spatial information from the trajectories of nearby robots during map prediction, and 3) a confidence network to measure the uncertainty of map prediction for effective exploration under resource constraints. We incorporate *4CNet* within our proposed robot exploration with map prediction architecture, *4CNet-E.* We then conduct extensive comparison studies with *4CNet-E* and state-of-the-art heuristic and learning methods to investigate both map prediction and exploration performance in environments consisting of uneven terrain and irregularly shaped obstacles. Results showed that *4CNet-E* obtained statistically significant higher prediction accuracy and area coverage with varying environment sizes, number of robots, energy budgets, and communication limitations. Real-world mobile robot experiments were performed and validated the feasibility and generalizability of *4CNet-E* for mobile robot map prediction and exploration.**

***Index Terms*—Mobile robot exploration, contrastive learning, map prediction, consistency models, irregular-shaped unknown environments**

I. INTRODUCTION

MOBILE robots can be deployed in unknown and resource-limited environments to complete a variety of tasks, including searching for victims in disaster scenes [1], [2], forest coverage [3], and planetary exploration [4], [5]. These environments are cluttered, leading to partial observability from limited onboard sensing [6], and may also be dynamic, with other robots pursuing their respective goals. Furthermore, mobile robots have limited onboard energy storage and communication capabilities [7]. For example, in disaster scenes, rubble can obstruct communication between robots, preventing the exchange of map information as this requires high-bandwidth connectivity. Similarly, in remote environments, the absence of charging stations and extreme conditions (i.e., harsh weather) can limit both energy usage and communication capabilities [8]–[10].

Existing mobile robots mainly use frontier-based exploration to achieve coverage in unknown environments [11]. Frontiers are selected by estimating the expected information gain from the robot's directly observed sensory data [12]. However, these estimations cannot account for the varying terrain and spatial structure in unobserved areas as they naively treat unknown regions as either entirely free-space or occupied [13]. As a result of this limited spatial awareness, a robot's exploration goals may become suboptimal, leading to redundant exploration efforts [14].

Robot map prediction can be used to predict the spatial configuration of unobserved regions in an unknown environment to help improve information gain during frontier selection [15]. Furthermore, map prediction assists with coverage of areas that are unreachable due to poor traversability and/or blocked pathways [16]. In scenarios, where mobile robots have limited energy budgets and communication capabilities, map prediction: 1) results in reduced travel, promoting energy conservation [17], and 2) minimizes the need for high-bandwidth communication of robot maps with other robots, thus alleviating the communication burden [18].

The robot map prediction problem for unknown environments with irregular obstacles and changing terrain can be complex due to non-repetitive environmental geometries and noisy sensory information. In addition, existing map prediction methods have only been used in static single robot environments and have not been extended to consider dynamic multi-robot scenarios in which other robots are present and completing their own goals. However, the trajectories of these other robots can provide implicit information about the spatial layout of unobserved regions [19]. Namely, a robot's deviation from a linear trajectory (indicating no direct route) characteristically implies the presence of an obstacle [20]. Therefore, a robot's navigation path within an unknown environment can reveal obstacle contours that have not been directly observed. To-date, single mobile robot map prediction methods have primarily been deployed in structured indoor

This work was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC), and in part by the Canada Research Chairs program (CRC). *(Corresponding author: Aaron Hao Tan).*
The authors are with the Autonomous Systems and Biomechatronics Laboratory (ASBLab), Department of Mechanical and Industrial Engineering, University of Toronto, Toronto, ON M5S 3G8, Canada (e-mail: aaronhao.tan@utoronto.ca; s.narasimhan@mail.utoronto.ca; nejat@mie.utoronto.ca).

environments, including office buildings where reoccurring spatial features such as corridors and rooms exist [21]–[25].

To facilitate robot map prediction of unknown environments with non-repeating irregular obstacles, consistency models can be considered. Consistency models are a new class of generative models that employ a multi-pass prediction approach [27]. They can be used to iteratively refine predictions to generate a single map prediction [26], [27], enhancing map accuracy. This approach can provide an advantage over existing single-pass autoencoder architectures, e.g., [14], [16], [28], and Generative Adversarial Networks (GANs), e.g., [29], [30], which also create map predictions in a single forward pass. Such approaches often produce less accurate maps due to the lack of iterative refinement in their prediction process [27]. To-date consistency models have not yet been developed for mobile robot map prediction.

In this paper, we present a novel robot exploration map prediction method called Confidence-Aware Contrastive Conditional Consistency Model (*4CNet*), to predict (*foresee)* unknown spatial configurations in unknown unstructured multi-robot environments with irregularly shaped obstacles. *4CNet* is incorporated within our proposed robot exploration with map prediction architecture, *4CNet-E,* consisting of a *Perception and Communication* subsystem and an *Exploration Planner*. *4CNet* uniquely integrates the three key components of confidence awareness, contrastive learning, and conditional consistency model for efficient robot exploration under limited energy budget and communication capabilities. The main contributions of *4CNet* are:

1) The development of a novel map prediction network that is the first to use a conditional consistency model to predict the spatial configuration of unobserved regions within a partially explored environment with irregularly shaped obstacles and uneven terrain.
2) The unique utilization of contrastive learning to pre-train a trajectory encoder for the extraction of spatial information from nearby robot trajectories; allowing our map prediction approach to account for both static and dynamic environment features.
3) The first implementation of a confidence network for map prediction to guide robots towards uncertain regions to maximize prediction accuracy with limited energy budgets.

II. Related Works

The review presented here is classified as: 1) mobile robot map prediction methods [14], [16], [17], [21]–[25], [28]–[32], and 2) robot energy-aware exploration methods [11], [33], [42], [43], [34]–[41].

A. Mobile Robot Map Prediction Methods

Map prediction methods can be classified into heuristic-based [21]–[25] or deep learning (DL)-based [14], [16], [17], [28]–[32] methods. Heuristic-based methods utilize predefined rules to interpret spatial layouts by exploiting structural patterns in the robot's environment. In contrast, DL methods utilizes deep neural networks to detect spatial features within a high-dimensional space in order to make map predictions.

1) Heuristic-based Methods

Heuristic map prediction methods have used either map databases (DB) [21]–[23], representative lines [24], or low-rank matrix completion (LRMC) [25] to define a set of predefined rules that have interpreted spatial layouts for the exploitation of structural patterns in the robot's environment.

In DB methods, individual robots have access to a database of 2D robot maps prior to deployment. These maps are typically of repetitive structured environments such as rooms and corridors [22]. In general, DB methods have two phases [21]–[23]. Firstly, they identify a reference map from the database based on structural similarities to the unknown region using metrics such as feature vectors from Fast Appearance Based Mapping (FabMAP2) [22], counts of overlapping occupied cells [21], or highest likelihood of feature resemblance [23]. Secondly, they merge a reference map with the unknown region using techniques such as RANSAC-based Voronoi graph alignment [22], spatial alignment using homogenous transform matrices [21], or Gaussian filtering for spatial coherence [23].

Representative lines methods have been used in rectilinear environments, where 2D lines represent straight walls and corners [24]. These methods extrapolate lines from observed areas to unobserved regions using an objective function to maximize: 1) wall count uniformity by balancing distribution of walls in the predicted room layout, and 2) an simplicity index to minimize the shape complexity of predicted room layouts.

LRMC methods have been used to reconstruct missing cell information in a robot 2D map matrix by exploiting the low rank and incoherence characteristics of an environment [25]. These methods utilize an iterative Singular Value Decomposition solver to minimize the norm of the matrix by aggregating the singular values [44]. Missing cells in the map matrix are predicted by exploiting the linear relationships between the columns and rows of the map matrix.

The aforementioned map prediction methods have been applied to robot exploration [21]–[23], and path planning for coverage [24], [25] problems. Namely, for exploration, the predicted robot maps have been used to complete the exploration of unobserved areas [21] and enhance frontier selection by providing expected information gain from unobserved parts of the environment [22], [23]. Experiments in structured indoor [21], [23] and repetitive underground tunnel environments [22] showed that exploration with map prediction provides more accurate maps and reduced travel distances when compared with non-map prediction exploration methods [45]. In coverage path planning, predicted maps inform offline methods such as Christofides [46] and tabu search [47]. Simulations in structured indoor [24] and grid world environments [25] have showed that path planning with map prediction has improved coverage ratio when compared to planning strategies without map prediction (i.e., lawnmower planning [48] and adaptive k-swap heuristic [49]).

2) Deep Learning-based Methods

DL based methods used for map prediction consist of autoencoder models [14], [16], [17], [28], [31], [32], or GANs [29], [30].

In an autoencoder model for robot map prediction, the encoder network uses down sampling layers to capture the spatial context from pixel-level features from either partially observed 2D maps [14], [16], [28], [32] or RGB-D images from robot-centric viewpoints [17], [31]. The decoder network then reconstructs the spatial embeddings extracted by the encoder network into map predictions through up sampling layers. Existing autoencoder models include network components such as either: 1) convolutional layers [28], or fully connected layers with skip connections [16], [17], [31], [32], in the encoder and decoder networks, or 2) a probabilistic latent space bottleneck [14] between the encoder and decoder networks.

In contrast, robot map prediction methods using GANs generate maps with a game-theoretic approach involving a generator and a discriminator convolutional network [29], [30]. The generator network learns to generate complete robot maps by replicating the distribution of robot maps that are present in a dataset. The discriminator network then evaluates the generated map predictions with maps in the dataset to provide an evaluation signal for the loss function of the generator during training. This adversarial process progressively refines the ability to produce robot maps [50].

The above DL based map prediction methods have been used for robot exploration [14], [16], [17], [28]–[30], [32], and semantic mapping [31] in simulated and real-world structured environments. For exploration, the predicted map from autoencoder models and GANs were used with information-theoretic frontier selection methods to improve exploration efficiency over exploration methods without map prediction in terms of travel distance [17], [32], [14], [30], coverage [14], [16], exploration time [28], and map accuracy [29]. For semantic mapping, the predicted map was used to segment and label different objects (i.e., chairs, desks) within the environment, in order to fill in missing geometric data within the map of a robot [31].

B. Energy-Aware Exploration Methods

Existing robot energy-aware exploration methods have focused on traversing unknown regions in an environment to generate maps, while minimizing travel distances and operation times [11], [51]. Maps or intended exploration goals were communicated between nearby robots in a team in a decentralized manner to select complementary goals in order to reduce redundant coverage [35]. These approaches can be categorized into three categories: 1) utility-based [11], [38]–[41], 2) market-based [33], [34], [42], [43], and 3) deep reinforcement learning (DRL) methods [35]–[37].

Utility-based robot exploration methods have used a utility function to select frontiers that maximize expected information gain and minimize travel cost [11], [38]–[41]. The objective is to maximize coverage and minimize robot energy consumption. On the other hand, market-based exploration methods have used a robot bidding procedure for frontiers based on estimated travel cost (distance, time) for exploration [33], [34], [42], [43]. Individual robots minimize their travel cost, while the team maximizes its coverage. DRL techniques such as Double Deep Q Recurrent Networks (DDRQN) [35], Multi-agent Proximal Policy Optimization (MAPPO) [36], and Multi-agent Deep Deterministic Policy Gradient (MADDPG) [37], have been used to train decentralized exploration policies that maximize the expected discounted reward over the horizon of an episode. These reward functions positively reward a robot for area coverage while negatively reward distance traveled and time elapsed. Robot experiments were conducted in both simulation and real-world structured environments, to highlight the difference in travel distance/time between exploration methods.

C. Summary of Limitations

Existing heuristic and DL-based map prediction methods can suffer from lower prediction accuracy in terms of spatial structural features and pixel-level image feature textures [32]. Namely, heuristic methods assume the existence of similar static environments with rectilinear spatial features during prediction [21]–[25]. Therefore, they are not able to provide accurate map predictions for unstructured environments with irregular-shaped objects and uneven terrain.

Both autoencoder models and GANs rely on single-pass predictions, where the input map is processed in a single forward (computational) pass through the network. This single-pass approach limits these DL methods from integrating intricate spatial features of complex environments into high-fidelity predictions, resulting in blurry maps with inconsistent structural details [27]. Furthermore, GANs have encountered unstable training and mode collapse due to the adversarial dynamics between generator and discriminator networks, resulting in poor performance in capturing diverse map data and environmental complexity during prediction [52], [53]. To-date, autoencoder models are favored for map prediction tasks for their ability to reconstruct robot maps from partial observations.

Existing map predictions methods also assume uniform confidence across all predicted pixels, which can lead to suboptimal decision-making during robot planning [54]. However, quantifying map prediction uncertainty allows more informed evaluation of potential information gain from visiting different exploration goals [32]. Furthermore, map prediction methods focus on single robot environments, not considering scenarios with multiple robots. In such scenarios, the trajectories of nearby robots can be uniquely leveraged to provide contextual data for unobserved regions [17].

On the other hand, energy-aware exploration methods do not address challenges where: 1) a robot may not have sufficient energy to explore an entire environment, and 2) a robot cannot share map data due to communication limitations. Therefore, a robot exploration method is needed that can utilize map prediction to maximize coverage, while addressing limitations in onboard energy storage and communication capabilities.

III. The Robot Map Prediction Problem For Resource Limited Exploration

A. Problem Definition

The robot map prediction problem for resource limited exploration requires a mobile robot to explore an unknown, dynamic, and unstructured environment. The environment may consist of other mobile robots achieving their own goals. Each robot explores the environment and operates under constrained communication capabilities and a limited energy budget. The goal is to maximize a robot's knowledge of the configuration of the environment using both the observed region of the environment, M_i^{obs}, and the predicted spatial configuration from the unobserved region of the environment, M_i^{pred}.

Robots: *N* number of non-holonomic mobile robots can exist in an environment, $R = \{r_1, r_2, \dots, r_n\}$. Each robot r_i has an onboard sensor with a sensing range of s for mapping its surroundings. The robots have their own individual exploration goals and only share their trajectory information when within this sensing range. Energy consumption, E_i, for robot r_i is modeled as a linearly decaying function with respect to the distance Δd_i traversed by the robot, and the change in elevation Δz_i during robot navigation. Therefore, the energy consumption model is defined as:

$$E_i(\Delta d_i, \Delta z_i) = q_i - \left[(w_i \cdot \Delta d_i) + \left(k_i \cdot \max(0, \Delta z_i)\right)\right], \tag{1}$$

where q_i and w_i are constants that denote the initial energy budget for r_i and the energy consumed per unit distance traveled, respectively. k_i is the energy cost for vertical motion along the *z*-axis. The function $\max(0, \Delta z_i)$ signifies that additional energy is consumed only when the robot is ascending ($\Delta z_i > 0$), while no additional energy is consumed for descending or traversing over flat terrain ($\Delta z_i \leq 0$). The trajectory of robot r_i is denoted as τ_i. This trajectory is a temporally ordered set of robot positions up to the current time step, t, and is represented as:

$$\tau_i = \left\{(x_{t_1}, y_{t_1}), (x_{t_2}, y_{t_2}), \dots, (x_t, y_t)\right\}. \tag{2}$$

Environment: The environment is represented by both traversable uneven terrain and non-traversable irregular shaped obstacles. The environment is discretized into a heightmap M_h, where each cell in the map, $m_{(x,y)}$, is characterized by its (x, y) coordinates and contains elevation information within a specified range $[l_{min}, l_{max}]$. Thus, the heightmap is defined as:

$$M_h = \left\{m_{(x,y)} \mid m_{(x,y)} \in [l_{min}, l_{max}], \forall (x, y)\right\}, \tag{3}$$

where each robot's position, $p_i = (x_i, y_i)$, represents a specific cell $m_{(x_i, y_i)}$ in M_h.

Communication between Robots: Robots can exchange their trajectories, τ_i, when they are within sensing range, s. The number of robots within s is defined as R_s; $R_s \subseteq R$. To account for the stochastic nature of communication in real-world environments, a Communication Success Probability (CSP), $C \in [0,1]$, [35], is incorporated to represent the likelihood of successful trajectory transmissions between robots. To implement CSP, each robot position at time t in the trajectory, τ_i^t, is associated with a Bernoulli event represented by a random variable B_t. B_t captures the success of communication: a value of 1 occurring with probability $\mathcal{q}$, denotes a successful transmission of τ_i^t; while a value of 0 with the complementary probability $1 - \mathcal{q}$, denotes a transmission failure. The transmitted trajectory, τ_i', is a subset of the robot's traversed trajectory τ_i. Specifically, τ_i' includes only those positions for which $B_t = 1$, and is expressed as:

$$\tau_i' = \{(x_t, y_t) \mid (x_t, y_t) \in \tau_i, B_t = 1\}. \tag{4}$$

Once r_i receives trajectory information from all other robots within sensing range, R_s, the communicated trajectories are aggregated into a collective set $\delta_t = \{\tau_i'\}_{i \in R_s}$.

Map Prediction Task: Each robot predicts the unexplored environment configuration based on 1) its own observed portion of the environment during exploration, M_i^{obs}, and 2) the trajectory information of nearby robots, δ_t. Thus, the predicted map $\widehat{M_\iota}$ of the entire environment is represented as:

$$\widehat{M_\iota} = f_\theta\left(\{M_i^{obs}, \delta_t\}\right). \tag{5}$$

$\widehat{M_\iota}$ is defined by $M_i^{obs} \cup M_i^{pred}$. The goal is to approximate the map prediction function f_θ such that $\widehat{M_\iota}$ can be used by a frontier-based exploration method, ∂, to account for both observed and predicted map information during exploration.

Exploration Objective Function: The exploration objective is to maximize robot spatial knowledge given a limited energy budget, q_i. Namely, the objective is to maximize the utility of a frontier, U, selected by ∂, over the time horizon, h, while adhering to energy consumption limits, E_i, of the robot r_i:

$$\begin{gathered}\text{maximize}\left[\sum_{j=1}^{\mathrm{h}} U\left(\partial\left(f_\theta\left(M_i^{obs}, \delta_t\right)\right)\right)\right], \\ s.t.\, E_i(\Delta d_i, \Delta z_i) \geq \left[(w_i \cdot \Delta d_i) + \left(k_i \cdot \max(0, \Delta z_i)\right)\right].\end{gathered} \tag{6}$$

B. Map Prediction Using Consistency Models

In order to learn the map prediction function f_θ in Eq. (5), we utilize consistency models [26]. Consistency models can predict the spatial layout within unexplored regions based on Gaussian noise input and nearby robot trajectories. This is achieved in two-stages: a noising stage and a denoising stage.

Noising Stage: A Probability Flow Ordinary Differential Equation (PF ODE) is used to model the temporal evolution of the map prediction process. Specifically, an PF ODE represents the transition from a robot's initial noiseless map state, $\mathfrak{m}_0$, to a terminal map state representing Gaussian noise, $\mathfrak{m}_T$. This transition occurs across a series of time steps $\mathcal{t}$, with each step undergoing a calculated perturbation by the addition of Gaussian noise. The output of the noising stage is a sequence $\{\mathfrak{m}_\mathcal{t}\}_{\mathcal{t} \in [\epsilon, T]}$, Fig. 1, where each step represents a map with gradual increase in noise. The noise applied at each $\mathcal{t}$ is determined by: 1) a predefined noise schedule that determines the standard deviation of the noise to be incrementally added, and 2) the total noising steps, T, to determine the number of increments required to reach full noise.

Denoising Stage: The map prediction function, f_θ, describes a consistency model. Its objective is to obtain the original map state $\mathfrak{m}_0$ from any map state $\mathfrak{m}_\mathcal{t}$ in the PF ODE sequence $\{\mathfrak{m}_\mathcal{t}\}_{\mathcal{t} \in [\epsilon, T]}$. Namely, the parameterized map prediction function considers the time step of the sequence $\mathcal{t}$, the partially known map M_i^{obs}, and the given robot trajectories δ_t:

$$\begin{gathered}f_\theta\left(M_i^{obs}, \mathcal{t}, \delta_t\right) = p_{\text{skip}}(\mathcal{t}) M_h + \\ p_{\text{out}}(\mathcal{t}) F_\theta\left(M_i^{obs}, \mathcal{t}, \delta_t\right),\end{gathered} \tag{7}$$

where p_{skip} and p_{out} are weighting coefficients that serve two purposes. First, given a time step near the origin of the PF ODE, denoted by $t < \epsilon$, f_θ provides the ground truth heightmap, M_h, as the output. Second, as t increases, the output of f_θ increasingly transitions to the output of the predictive model, F_θ. The change in output between M_h and F_θ is achieved by enabling $p_{\text{skip}}(t)$ to decrease and $p_{\text{out}}(t)$ to increase as t increases. The initial conditions are $p_{\text{skip}}(\epsilon) = 1$ and $p_{\text{out}}(\epsilon) = 0$. Thus, p_{skip} and p_{out} ensure that f_θ is differentiable for model training through backpropagation. During inference, f_θ can be sampled with multiple passes through the map prediction network to refine the prediction quality iteratively.

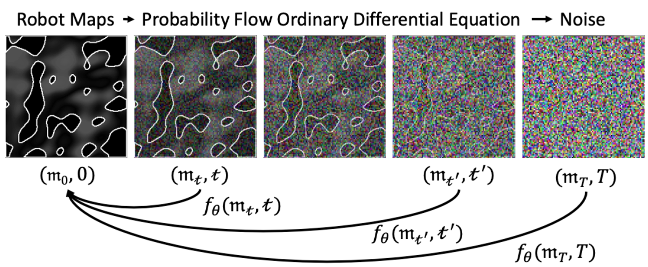


Fig. 1. Map prediction function, f_θ, described as a consistency model that learns to transform any robot heightmap state, $\mathfrak{m}_t$, along the Probability Flow Ordinary Differential Equation sequence to the original map state, $\mathfrak{m}_0$. In the original map state at $\mathfrak{m}_0$, white lines represent obstacle contours, and the gray scale gradient represents traversable uneven terrain. $\mathfrak{m}_T$ represents the terminal map state consisting of Gaussian noise.

IV. The Robot Exploration Architecture with Map Prediction

The proposed DL robot exploration with map prediction architecture, *4CNet-E*, Fig. 2, has been developed to predict maps of partially explored environments consisting of irregular shaped obstacles and uneven terrain. The goal of this architecture is to use the predicted maps to guide robots towards unexplored regions with high expected information gain, while addressing fixed energy budgets and limited communication. It consists of three main subsystems: 1) *Perception and Communication,* 2) *4CNet*, and 3) *Exploration Planner. The Perception and Communication* subsystem generates a 2D partial map, M_i^{obs}, using robot LiDAR and odometry information via the *Simultaneous Mapping and Localization (SLAM)* module. It also communicates and receives trajectory information, δ_t, from nearby robots using the *Communication* module. M_i^{obs} and δ_t are used by the *Trajectory Encoder* module in *4CNet* to produce trajectory embeddings, σ_{traj}, that extract spatial features from δ_t to condition the *Map Prediction Network* (*MPN*). The *MPN* uses these embeddings to generate the predicted map $\widehat{M_\iota}$. $\widehat{M_\iota}$ is used by the *Confidence Network* (*CN*) to obtain a confidence map C_i for each predicted pixel within $\widehat{M_\iota}$. The output of *4CNet* is both $\widehat{M_\iota}$ and C_i, which are used in the *Exploration Planner* subsystem to select a frontier goal location for the robot using a utility function U. The selected frontier goal is used by the *Navigation Controller* to generate a navigation trajectory, τ_{goal}. The main subsystems of *4CNet-E* are discussed below in further details.

A. Perception and Communication Subsystem

The SLAM module in the *Perception and Communication subsystem* utilizes robot odometry, ρ_i, and onboard 3D LiDAR observations, o_i, to localize the robot in the environment, p_i, and to generate a map of the observed regions, M_i^{obs}. Real-time Appearance Based Mapping (RTAB-Map) [55] is used to generate a graph-based map with nodes representing LiDAR scans. These nodes are connected by edges that represent spatial relationships between successive LiDAR scans. The *Communication* module is used to transmit the robot's own trajectory τ_i', and receive trajectory data, δ_t, from other robots within range s. δ_t is represented as a 2D binary array, producing a binary trajectory image, $\widehat{\delta_t}$, that visually represents the trajectories of the robots. Both $\widehat{\delta_t}$, and M_i^{obs} are provided to *4CNet* for map prediction.

B. 4CNet

4CNet consists of three modules: 1) *Trajectory Encoder*, 2) *Map Prediction Network*, and 3) *Confidence Network.*

1) Trajectory Encoder

The objective of the *Trajectory Encoder* module is to encode the available trajectories from other robots, δ_t, into a trajectory embedding vector, σ_{traj}. We introduce a unique *Contrastive Map Trajectory Pretraining (CMTP)* framework to contrast robot trajectory and map features in order to capture spatial information (i.e., obstacle contours) implicitly present in robot trajectories. The *CMTP* framework utilizes two distinct encoders: a map encoder, E_{map}, for extracting spatial features from M_h used during pretraining, and a trajectory encoder, E_{traj}, for extracting robot coordinate features from $\widehat{\delta_t}$, used during both pretraining and inference. Both encoders use a ResNet50 backbone [56], and a projection head with two fully connected (FC) layers, Fig. 2. During contrastive learning [57], E_{map} and E_{traj} are pretrained simultaneously to align robot spatial and coordinate features from M_h and $\widehat{\delta_t}$ into a common representation space, $\mathcal{R}$. Specifically, E_{map} transforms the ground truth heightmap M_h into a map embedding vector, σ_{map}. Concurrently, the trajectory image, $\widehat{\delta_t}$, is used by E_{traj} to generate a robot trajectory embedding vector, σ_{traj}. The contrastive loss function, $\mathcal{L}_{CMPT}$, for *CMTP* is defined as:

$$\mathcal{L}_{CMTP} = -\mathbb{E}\left[\log \frac{\exp\left(\mathscr{k}\left(\sigma_{map}, \sigma_{traj}\right)/\mu\right)}{\sum_{\sigma'_{traj} \in \mathcal{D}_{MT}} \exp\left(\mathscr{k}\left(\sigma_{map}, \sigma'_{traj}\right)/\mu\right)}\right], \tag{8}$$

where $\mathcal{L}_{CMTP}$ computes the expected value of the negative log probability that a robot heightmap embedding, σ_{map}, has a higher similarity score with its corresponding trajectory embedding, σ_{traj}, compared to other trajectory embeddings, σ'_{traj}, within the dataset, $\mathcal{D}_{MT}$. Herein, $\mathcal{D}_{MT}$, is a dataset of robot heightmaps and trajectories. The function, $\mathscr{k}$, measures the cosine similarity between embedding pairs, outputting a similarity score. The exponential function, exp, is used to normalize similarity scores for effective gradient descent optimization during training. The temperature parameter, μ, is used to control the sharpness of the distribution of similarity

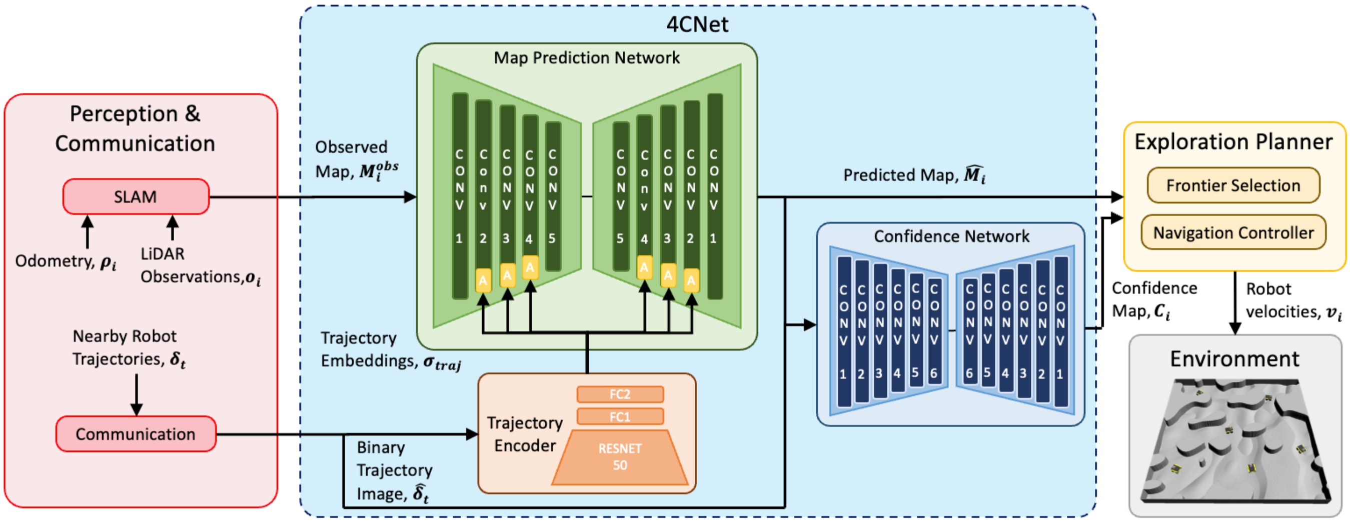


Fig. 2. The *4CNet-E* exploration with map prediction architecture for resource limited exploration in 3D unknown unstructured multi-robot environments with uneven terrain. **A** in the *Map Prediction Network* denotes the cross-attention mechanism for robot trajectory conditioning. **CONV** represents Convolutional Blocks, and **FC** represent Fully Connected layers.

scores to directly influence the gradient magnitudes during backpropagation. Minimizing $\mathcal{L}_{CMTP}$ will maximize the similarity scores between map-trajectory embedding pairs. The *Trajectory Encoder* provides trajectory embeddings, σ_{traj}, to the *Map Prediction Network*.

2) Map Prediction Network

The *Map Prediction Network* (*MPN*) predicts the spatial configuration of an unobserved region within a partially explored environment using: 1) the robot observed map M_i^{obs}, and 2) the trajectory embeddings σ_{traj}. The *MPN* is designed as a 2D U-Net architecture, where the encoder consists of five convolutional blocks with output channels of [128, 128, 256, 256, 512], Fig. 2. The decoder mirrors the encoder by having an equal number of convolutional blocks with the reverse output channels [512, 256, 256, 128, 128]. Cross-attention is integrated into the middle three blocks of both the encoder and decoder in the *MPN* to only integrate relevant trajectory features during the map prediction process.

For training of the *MPN*, an online network, $\boldsymbol{f_\theta}$, and a target network, $\boldsymbol{f_{\theta^-}}$, are used, where $\boldsymbol{f_\theta}$ is used to predict the robot heightmap state at $t+1$, and $\boldsymbol{f_{\theta^-}}$ is used to predict the robot heightmap state at t, along the PF ODE sequence. To ensure stable training, the weights of $\boldsymbol{f_{\theta^-}}$ are set as an exponential moving average (EMA) of the weights of $\boldsymbol{f_\theta}$, described by:

$$\theta^- = \mathcal{H}\theta^- + (1-\mathcal{H})\theta, \tag{9}$$

where $\mathcal{H}$ is the smoothing factor. EMA is used for θ^- to prevent drastic shifts in the $\boldsymbol{f_{\theta^-}}$ behavior due to large updates to the network parameters [58]. The *MPN* loss function, $\mathcal{L}_{MPN}$, uses stochastic gradient descent to minimize the prediction differences between the two networks:

$$\mathcal{L}_{MPN} = \mathbb{E}\left[\lambda(t_n)\mathcal{L}_m\left(\begin{matrix} f_\theta\left(\mathfrak{m}_{t_{n+1}} + t_{n+1}\xi, t_{n+1}, \tau_i\right), \\ f_{\theta^-}\left(\mathfrak{m}_{t_n} + t_n\xi, t_n, \tau_i\right)\end{matrix}\right)\right]. \tag{10}$$

Herein, $\mathcal{L}_{\mathrm{m}}$ computes the difference between the predicted maps from f_θ and f_{θ^-}, while adhering to the consistency property in Eq. (7). The weighting function $\lambda(t_n)$ adjusts the significance of each term in the loss. ξ is the Gaussian noise vector and represents the normally distributed noise sampled from $N(0,\mathrm{I})$ [26]. We designed $\mathcal{L}_m$ as a compound loss function that uses Perceptual Image Patch Similarity (LPIPS) [59] and the Edge Loss function, $\mathcal{L}_e$. In particular, LPIPS is used to evaluate the perceptual similarity score between the terrain features in the predicted maps, $(\mathfrak{m}_\theta, \mathfrak{m}_{\theta^-})$, from f_θ and f_{θ^-}, respectively. The goal is to ensure that both predicted maps have high similarity in terms of visual and structural characteristics. The LPIPS function, $\mathrm{LPIPS}(\mathfrak{m}_\theta, \mathfrak{m}_{\theta^-})$, is expressed as [59]:

$$\mathrm{LPIPS}(\mathfrak{m}_\theta, \mathfrak{m}_{\theta^-}) = \phi\left(\left\|V(\mathfrak{m}_{\theta_i}) - V(\mathfrak{m}_{\theta^-_i})\right\|_2^2\right), \tag{11}$$

where V represents a function that extracts pixel features from the predicted maps, $\mathfrak{m}_\theta, \mathfrak{m}_{\theta^-}$. The function, ϕ, applies a nonlinear transformation to the squared Eclidean distance between these extracted pixel features. This transformation converts the differences in the pixel feature space into the perceptual similarity score [59].

The Edge Loss function, $\mathcal{L}_e$, is used to compare boundary contours of irregularly shaped obstacles within the predicted robot heightmaps $(\mathfrak{m}_\theta, \mathfrak{m}_{\theta^-})$, and is formulated as:

$$\mathcal{L}_e(\mathfrak{m}_\theta, \mathfrak{m}_{\theta^-}) = \frac{1}{\aleph}\sum_{i=1}^{\aleph} \left\| \mathscr{s}(\mathfrak{m}_{\theta_i}) - \mathscr{s}(\mathfrak{m}_{\theta^-_i}) \right\|_2^2, \tag{12}$$

where $\mathscr{s}$ is the Sobel operator [60] used to detect the boundary contours of obstacles in $\mathfrak{m}_\theta$, and $\mathfrak{m}_{\theta^-}$. By combining LPIPS, and $\mathcal{L}_e$, $\mathcal{L}_m$ is formulated as:

$$\begin{aligned}\mathcal{L}_m = {} & \mathscr{a} \cdot \mathrm{LPIPS}(\mathfrak{m}_\theta, \mathfrak{m}_{\theta^-}) \\ & + \mathscr{b} \cdot \mathcal{L}_e(\mathfrak{m}_\theta, \mathfrak{m}_{\theta^-}),\end{aligned} \tag{13}$$

where a and b are scaling hyperparameters for the LPIPS and $\mathcal{L}_e$ components, respectively. The output of the *MPN* is the predicted robot heightmap $\widehat{M_\iota}$ which is provided to both the *Confidence Network* and *Exploration Planner* modules.

3) Confidence Network

The *Confidence Network (CN)* is used to measure the uncertainty of $\widehat{M_\iota}$. The aim of *CN* is to guide robot exploration towards frontier regions with higher prediction uncertainty. Inputs into the *CN* include a single two channel image, where the first channel contains the predicted map $\widehat{M_\iota}$, and the second channel contains the nearby robot trajectory image $\widehat{\delta_t}$.

The *CN* uses a Residual Fully Convolutional Variational Autoencoder (RFC-AEM) model to produce a confidence map, C_i, which represents the uncertainty of each predicted cell $m_{(x,y)}$ within $\widehat{M_\iota}$. The RFC-AEM consists of an encoder and decoder, Fig. 2. The encoder includes six convolutional layers, each followed by batch normalization and a Leaky ReLU activation function. These layers facilitate map feature extraction and utilize a latent space bottleneck for dimensionality reduction to capture only the most salient features from the input map. The decoder mirrors the encoder with six transposed convolutional layers, enabling the reconstruction of detailed confidence maps from the compact encoded latent representations of the encoder.

The *CN* adopts a self-supervised training paradigm. Ground truth confidence maps, C_i^{GT}, are obtained by comparing the prediction, $\widehat{M_\iota}$, with the ground truth heightmap, M_h, where each value in C_i^{GT} is a binary indicator, specifying whether the prediction is correct or incorrect. To train the *CN*, a pixel-wise Mean Squared Error (MSE) loss function is utilized:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{\aleph}\sum_{x,y\in\widehat{M_\iota}} \left(C_i(x,y) - C_i^{GT}(x,y)\right)^2, \tag{14}$$

where $\aleph$ represents the total number of pixels in $\widehat{M_\iota}$, and x, y are the pixel coordinates of $\widehat{M_\iota}$. Minimizing $\mathcal{L}_{\text{MSE}}$, minimizes the uncertainty measure between $\widehat{M_\iota}$ and M_h. The predicted confidence map C_i is provided to the *Exploration Planner* for frontier exploration.

C. Exploration Planner

The objective of the *Exploration Planner* subsystem is to: 1) choose frontiers that maximize the utility function, U, while maintaining the energy budget of a robot, Eq. (6), using the *Frontier Selection* module, and 2) have the robot navigate towards the frontier goal using the *Navigation Controller*. The *Frontier Selection* module, ∂, uses the robot's predicted map $\widehat{M_\iota}$ and the associated confidence map C_i to evaluate the utility, u, of a frontier location, $g(x,y)$, and selecting the g with the highest u. The utility function, $U(g)$, calculates the u of each g based on the expected information gain I, traversability score $\mathcal{T}$, and travel distance to goal D:

$$U(g) = \alpha \times I + \beta \times \mathcal{T} + \gamma \times D. \tag{15}$$

Herein, the expected information gain is determined by the estimated traversable area and its uncertainty. Specifically, I is computed by evaluating the traversable areas, A_{d_r}, in the predicted map, $\widehat{M_\iota}$, within a fixed radius, d_r, around the frontier position, g. Note, only the predicted region, M_i^{pred}, within $\widehat{M_\iota}$ is considered. This evaluation involves integrating the traversable area values in $\widehat{M}_l$ which are weighted by the corresponding confidence scores from C_i over A_{d_r}. I is then obtained by averaging $\int_{A_{d_r}} \widehat{M}_l \cdot C_i \, dA$ over the total area of A_{d_r}:

$$I = \frac{1}{|A_{d_r}|}\int_{A_{d_r}} \widehat{M}_l \cdot C_i \, dA. \tag{16}$$

The traversability score $\mathcal{T}$ is the summation of elevation values $m_{(x,y)}$ along the shortest collision-free path between the robot's current position p and the frontier g denoted as $\tau_{p\to g}$. Thus, the traversability score is described by:

$$\mathcal{T} = \sum_{\tau_{p\to g}} m_{(x,y)}. \tag{17}$$

D is calculated as the length of $\tau_{p\to g}$. The coefficients $[\alpha, \beta, \gamma]$ in Eq. (15) are determined through domain expert tuning in order to prioritize frontiers that maximize I while minimizing $\mathcal{T}$ and D. The frontier g with the highest u is used by the *Navigation Controller* module for global planning using Spatial Temporal A* [61] and local path planning using Timed Elastic Band Planner [62].

V. Datasets

We developed two simulated datasets to train *4CNet.* These datasets include: 1) a map-trajectory dataset for training the *Trajectory Encoder* and *MPN* modules, and 2) a prediction map dataset used for the training of the *CN*.

Map-Trajectory Dataset, $\mathcal{D}_{\mathbf{MT}}$: 2D heightmaps, M_h, were generated using the Diamond-Square Algorithm [63], Fig. 3(a), with irregularly shaped obstacles (white lines), and uneven terrain (grayscale gradients). Each heightmap has a resolution of 224 × 224 pixels which represents a spatial area of 30 × 30 m. The number of robots in each environment is randomly varied from 2 to 6 robots, with each robot having random feasible start and end positions in the environment. The A* search algorithm [64] was used to generate collision-free paths between these positions for each heightmap. To simulate communication dropout, CSP was applied to probabilistically remove robot positions from the A* generated trajectories at dropout rates of 25%, 50%, and 100% (Eq. (4)), Fig 3(c)-(e). In total, the dataset includes 75,000 pairs of heightmaps, M_h, and robot trajectories, δ_t. For evaluation purposes, 20% of the $\mathcal{D}_{\text{MT}}$ was used as the test set, $\mathcal{D}_{\text{MT}-\text{T}}$.

Predicted Map Dataset, $\mathcal{D}_{\mathbf{PM}}$: The Predicted Map Dataset contains 20,000 samples of: 1) 2D predicted heightmaps, $\widehat{M_\iota}$, 2) ground truth heightmaps, M_h, and 3) corresponding robot trajectories, δ_i. The predicted heightmaps were generated by randomly masking regions of the ground truth heightmaps in the $\mathcal{D}_{\text{MT}}$ to simulate partial maps (Fig. 3(b)) and using the *MPN* to generate the predicted map $\widehat{M_\iota}$. Each $\widehat{M_\iota}$ is paired with its M_h as well as the corresponding δ_i from the $\mathcal{D}_{\text{MT}}$.

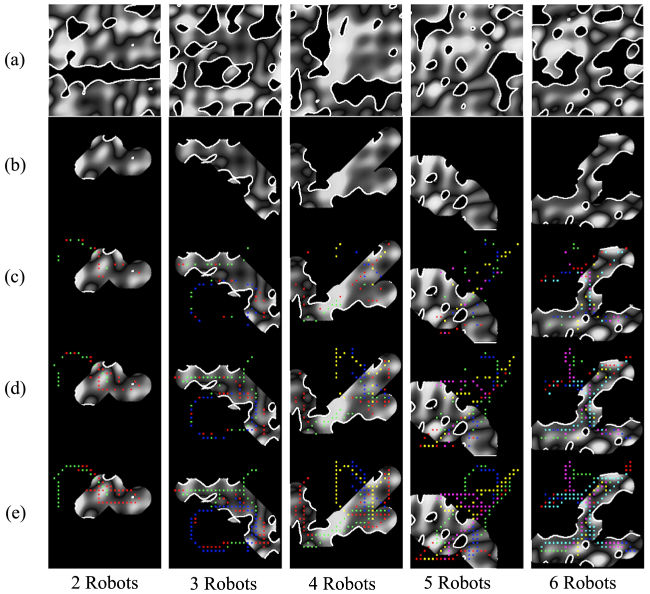


Fig. 3. Robot heightmaps for scenes with 2 to 6 robots: (a) Ground truth map of environment with irregularly shaped obstacles (white lines) and uneven terrain (grayscale gradient for terrain height); and (b) Input map of observed area, unobserved (masked) regions are black. The dotted color lines represent communicated trajectories from different robots at: (c) 25% CSP, (d) 50% CSP, and (e) 100% CSP.

VI. Training of 4CNet

Training consisted of: 1) using the proposed *CMTP* framework to train the *Trajectory Encoder*, 2) training of the *Map Prediction Network* conditioned on robot trajectory embeddings, and 3) training of the *Confidence Network* based on the predictions of the *MPN*. All training was conducted on a workstation with i9-13900KF Intel CPU, Nvidia RTX 4090 GPU, and 64 GB of RAM.

A. Contrastive Map Trajectory Pretraining

The *CMTP* framework was used to train the Trajectory and Map Encoder modules in parallel, as described in Section IV.B.1. Namely, *CMTP* was trained with a batch size of 128. A neural dropout rate of 0.3 was used to promote generalizability of the model and minimize overfitting [65]. We used a learning rate of 0.001 for gradient descent optimization, and a temperature of 1 was selected for the softmax function in the contrastive loss, Eq. (8). The *CMTP* was trained for 10 hours over a duration of 56 epochs, Fig. 4(a). We implemented early stopping and obtained the lowest validation loss at the 15th epoch (approximately 2.5 hours of training).

B. Map Prediction Network Training

The *Map Prediction Network* was trained with a batch size of 8. The hyperparameters of the minimum and maximum standard deviations for the PF ODE noise were set to 0.002 and 80, similar to [26]. A Kerras schedule hyperparameter of 7 was used with initial and final PF ODE time steps of 2 and 100. The initial exponential moving average decay rate of 0.95 was used for the target network $\boldsymbol{f_{\theta^-}}$, and a learning rate of 0.00002 was utilized during training [26]. The scaling factors a and b from Eq. (13) were set to 0.1 and 0.04, respectively. The *MPN* was trained for a total of 100,000 steps over 37 hours. The training loss graph is presented in Fig. 4(b). The loss converged to 0.01 by 100,000 training epochs.

C. Confidence Network Training

The *Confidence Network* was trained with a batch size of 64 and a learning rate of 0.0001. The training loss converged to 2.0×10^6 within 100 epochs after 4.5 hours of training, Fig. 3(c).

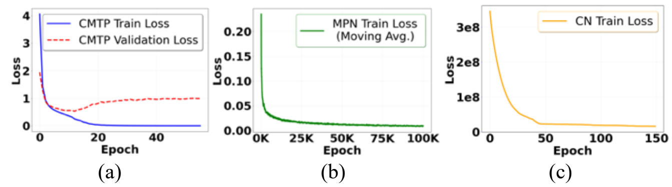


Fig. 4. (a) *CMTP* training and validation losses, (b) *MPN* training loss, and (c) *CN* training loss.

VII. Simulated Experiments

We conducted two simulated experiments to investigate the map prediction performance of *4CNet-E* for resource-limited exploration in complex unstructured environments. We set the total number of denoising time steps, t_{total}, as 30, to enable multi-pass prediction using *4CNet*; thereby, allowing iterative refinement of the generated map during robot map prediction. The first experiment compares map prediction accuracy of our *4CNet* method and state-of-the-art (SOTA) heuristic and DL methods with varying CSPs, and number of robots in the environment. The second experiment provides a comparison of robot coverage during resource-limited exploration between *4CNet-E* and other exploration methods.

A. Comparison Study for Map Prediction in Cluttered and Unknown Environments

A comparison study was performed to evaluate our *4CNet* subsystem against SOTA heuristic and learning methods for different number of robots and CSP. We measured map prediction performance using the following metrics: 1) MSE for overall prediction error; 2) Obstacle Intersection over Union (O-IOU) to measure only the ratio of overlapping obstacle pixels between the predicted map and the ground truth map in order to evaluate the accuracy of predicted obstacle locations and shapes; 3) Feature Similarity Index (FSIM) [66] to assess the structural and feature similarity of predicted and ground truth heightmaps; and 4) Valid Trajectory Score (VTS), a metric we created to measure the proportion of a robot's trajectory that is on traversable terrain relative to the entire trajectory in the predicted heightmap: VTS $= \tau_i^v/\tau_i$. τ_i^v denotes the sequence of robot positions within a robot trajectory that coincides with traversable terrain (non-obstacle space).

1) Comparison Methods: We compared *4CNet* with three SOTA methods.

1. Database-based method (DB) [23]**:** The DB method is a heuristic-based approach which selects a reference heightmap from a database based on feature resemblance. The reference map is then integrated into the unexplored target area using Gaussian filtering to create a predicted robot heightmap, Fig. 5(a). The DB method is the only heuristic-based method that can address map prediction in unstructured and rough terrain environments through the use of a database containing heightmaps from similar environments.

2. Autoencoder Model method (AEM) [14]**:** The AEM method is a DL approach consisting of encoder and decoder networks, Fig. 5(b). Both networks utilize a ResNet50 architecture with skip connections [14]. The input to the AEM is a two-channel image: the first channel contains the grayscale partially observed robot heightmap, and the second channel consists of the mask of the target region. The mask has dimensions of 80×80 pixels, the same as in [14]. The encoder network performs down sampling of the partially complete robot heightmap to capture spatial features. The decoder network reconstructs these spatial features from latent variables obtained from the encoder network, producing a predicted map. To achieve prediction of the entire unknown region, a sliding window technique is implemented. Predictions are then made in a cascading manner. The final output of AEM is a 2D grayscale robot heightmap with predicted spatial configuration of the unknown region. The AEM method was selected as it is the only DL method that can generate diverse map predictions for unstructured environments, without suffering from mode collapse.

3. Trajectory Conditioned AEM method (T-AEM): We extended the AEM method to incorporate robot trajectory data for map prediction, i.e., T-AEM, as to the authors' knowledge, trajectory conditioned map prediction methods do not currently exist. In T-AEM, the inputs are the same as the AEM approach with the addition of robot trajectory embeddings within the latent space, Fig. 5(c). Namely, we utilized the *Trajectory Encoder* module from our proposed *4CNet* model to allow T-AEM to condition its map predictions on nearby robot trajectories, similar to *4CNet*. The T-AEM is used as a benchmark to evaluate the impact of adding trajectory information to a current SOTA method.

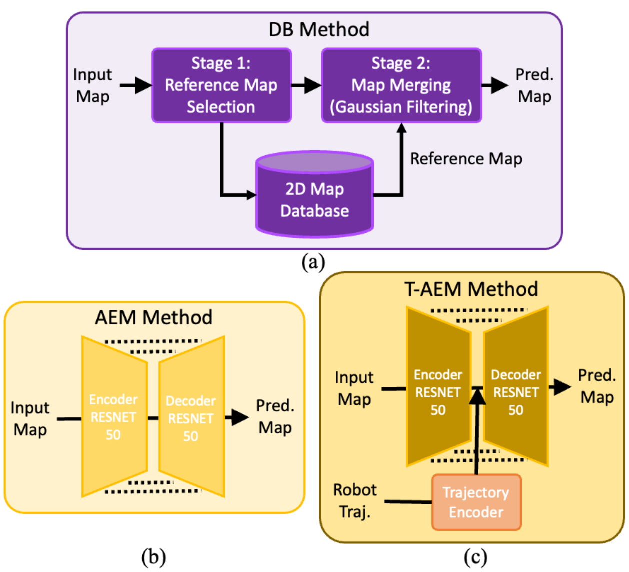


Fig. 5. (a) Database-based (DB) method, (b) Autoencoder model (AEM) method, and (c) Trajectory conditioned AEM method (T-AEM).

For the DB method, we randomly selected 5,000 heightmaps from the $\mathcal{D}_{\text{MT}}$ database (the same number of maps as in [23]) during each prediction, to balance reference map quality and search speed. Both AEM and T-AEM were trained on $\mathcal{D}_{\text{MT}}$.

2) Comparison Results: Table I and Figure 6 provide a comparative analysis of our *4CNet* and the SOTA map prediction methods. Each prediction method used as input a masked heightmap from $\mathcal{D}_{\text{MT}-\text{T}}$, i.e., Fig. 6(a). In general, our *4CNet* achieved the lowest average MSE and highest average O-IOU, FSIM and VTS at each CSP level across the number of robots. The DB method had the lowest performance among the methods. This was due to its reliance on a priori reference maps in its dataset to accurately represent an unknown environment. Therefore, when new obstacles that were not represented in the database were observed, the map prediction from DB resulted in misaligned and disjointed obstacle prediction. Furthermore, the use of Gaussian blurring to integrate these misaligned maps with actual observed maps resulted in an overall blurred predicted map, as shown in Fig. 6(b).

The AEM method also had higher MSE and lower O-IOU, FSIM and VTS than *4CNet*. In particular, the lower O-IOU score of 0.24 compared to *4CNet* (0.91 with 100% CSP) was due to AEM not being able to accurately reconstruct obstacles, leading to fragmented and pixelated predictions, Fig. 6(c). This fragmentation was primarily due to: 1) the single-pass prediction of AEM, and 2) the fixed dimension target region (e.g., 80×80 pixels) of AEM, that required a sliding window prediction technique to condition subsequent predictions based on prior predictions, resulting in cascading pixel errors.

As the T-AEM method incorporated robot trajectories during map prediction, it had a statistically significant improvement as defined by Friedman tests ($p < 0.001$), in terms of MSE, O-IOU and FSIM when compared to AEM with 100% CSP. This showed robot trajectories improved heightmap prediction. However, it is interesting to note that since T-AEM had similar VTS to AEM, the robot trajectory embeddings did not seem to improve the obstacle prediction accuracy of T-AEM. This is due to only a portion of the communicated robot trajectory being considered within each fixed dimension prediction window, leading to the incomplete and fragmented obstacle predictions by T-AEM, Fig. 6(d).

The better performance of *4CNet* was primarily due to the advantages of consistency models in: 1) multi-pass prediction, and 2) the use of arbitrarily shaped and sized target regions during prediction, which were not limited to a fixed dimension. Namely, the multi-pass prediction allowed for refinement of the map over a sequence of denoising time steps, resulting in predictions, Fig. 6(e), that closely matched ground the ground truth height maps, Fig. 6(f). This was further supported by *4CNet* having the lowest MSE, and highest FSIM. As *4CNet* was able to account for varying target regions and thus, effectively considered the entire robot trajectories communicated during each prediction. Furthermore, *4CNet* having the highest O-IOU and VTS was due to its ability to better predict obstacles in the environment, Fig. 6(e).

With respect to CSPs, *4CNet* achieved lower MSE, and higher O-IOU, FSIM, and VTS compared to T-AEM across all CSPs. Friedman Tests conducted across the MSE, O-IOU, FSIM and VTS metrics for the AEM and DB methods and all CSPs for the *4CNet* and T-AEM methods found statistically significant differences existed ($p < 0.001$). Post-hoc Wilcoxon Signed-rank tests with a Bonferroni correction confirmed a statistically significant difference in all four metrics when *4CNet* was compared with each SOTA method, across all CSPs ($p < 0.0167$). Therefore, validating *4CNet*'s improved performance over the other methods.

TABLE I
MAP PREDICTION PERFORMANCE COMPARISON

Method	CSP	# of Robots	MSE↓	O-IOU ↑	FSIM↑	VTS↑
DB	-	1	78.47	0.29	0.36	0.15
AEM	-	1	69.18	0.24	0.41	0.46
T-AEM	25%	2-6	74.53	0.22	0.40	0.45
	50%	2-6	67.37	0.26	0.43	0.48
	100%	2-6	64.97	0.28	0.44	0.47
4CNet (ours)	25%	2-6	56.23	0.52	0.49	0.82
	50%	2-6	53.20	0.58	0.53	0.89
	100%	2-6	50.35	0.59	0.55	0.91

↑ indicates a higher value represents better performance.
↓ indicates a lower value represents better performance.

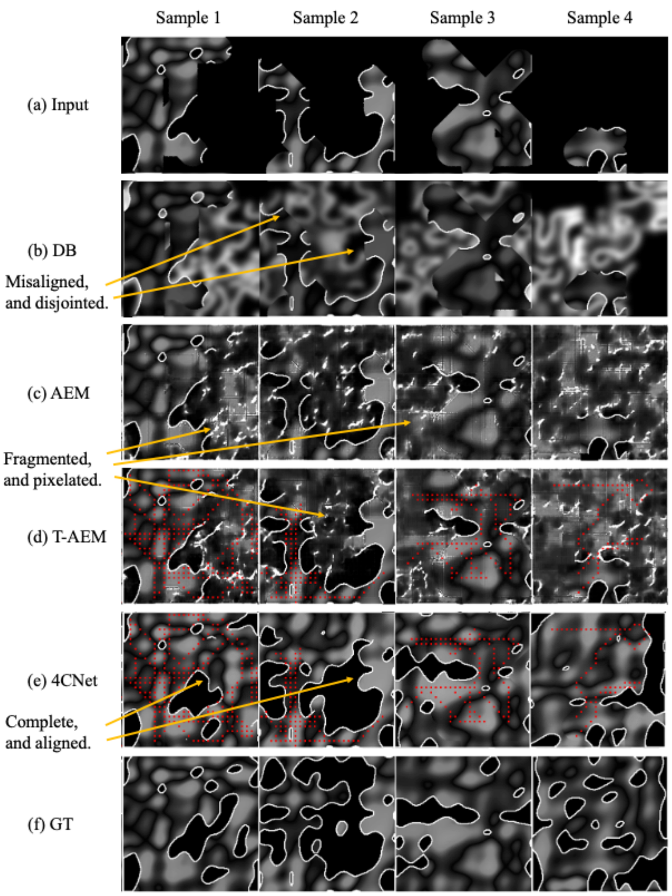


Fig. 6. Predicted map results using the: (a) partially completed maps from $\mathcal{D}_{\mathbf{MT-T}}$ as input for the: (b) DB, (c) AEM, (d) T-AEM, and (e) *4CNet* methods, compared to (f) ground truth (GT) maps. Red dotted lines denote communicated robot trajectories.

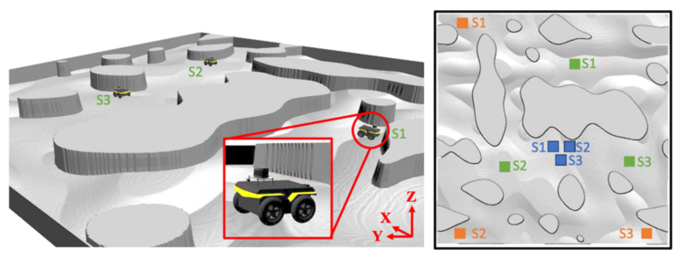
B. *Comparison Study for Exploration with Map Prediction in Resource Limited 3D Environments*

We evaluated the performance of *4CNet-E* in 3D simulated resource limited environments with uneven terrain and irregularly shaped obstacles. We introduced two environment sizes and three energy budgets. Four robot exploration methods were compared with *4CNet-E*: namely, exploration methods with 1) no map prediction, 2) AEM map prediction, 3) T-AEM map prediction, and 4) *4CNet* without CN map prediction. We measured the percentage of area coverage for each energy budget.

1) Mobile Robots: Three Clearpath Jackal mobile robots were used, Fig. 7(a). Each robot was equipped with a 360-degree LiDAR with a sensing range of 1.5 m, and both wheel encoders and an inertial measurement unit (IMU). Communication of robot trajectories was facilitated when the robots were within the aforementioned sensing range using 100% CSP.

2) Frontier Selection: Frontiers were selected using the utility equation in Eq. (15). The coefficients were set to [4, -1, -5] based on an expert-guided search, with the aim to maximize I while minimize $\mathcal{T}$ and D.

3) Environment: Two 3D environments were randomly generated using ROS Gazebo, consisting of uneven traversable terrain and irregularly shaped non-traversable obstacles, Fig. 7(a). The sizes of these environments were $15\ \text{m} \times 15\ \text{m}$ ($225\ \text{m}^2$) and $30\ \text{m} \times 30\ \text{m}$ ($900\ \text{m}^2$). The elevation of the traversable terrain ranged from 0 to 0.3 m to represent uneven slopes for the Clearpath Jackal robots to traverse. Obstacle heights were 0.7 m in the *z*-axis and were not traversable. Three sets of initial robot positions were used as shown in Fig. 7(b): where robots started 1) at opposite ends of the environment, 2) in the center nearby each other, and 3) random locations with at least a minimum distance of 3 meters.



(a) (b)

Fig. 7. (a) Robots in 3D simulation environment with irregularly shaped obstacles and uneven terrain. (b) Three sets of initial positions for each of the 3 mobile robots. S1-3 denote the starting positions of each robot.

4) Energy Budgets: Robots operated under three distinct energy budget levels: Low, Medium, and High. These energy budgets allowed the robots to only explore a portion of an environment, promoting the use of map prediction in the exploration process. The energy budgets were defined empirically in terms of area coverage in square meters, A_c, as a function of maximum travel distances in meters, d_{max}, and total area of the environment, A:

$$A_c = 0.0615({d_{max}}^{0.985}/A^{-0.635})\,. \quad (18)$$

Using Eq. (18), energy budgets of 25 m, 50 m, and 75 m were chosen for the Low, Medium, and High levels in the $225\ \text{m}^2$ environment size and 40 m for Low, 85 m for Medium, and 125 m for High in the larger $900\ \text{m}^2$ environment size.

5) Comparison Methods: We compared our *4CNet* exploration method (*4CNet-E*) against both non-predictive and predictive exploration methods. All exploration methods utilized the same frontier selection approach, Eq. (15), in order to directly compare performance in terms of map prediction influence on exploration. Furthermore, the comparison methods utilized a uniform confidence score in estimating the expected information gain, as they do not have a method for predicting the map prediction uncertainty. Thus, each predicted map pixel is assigned the same score for frontier selection.

1. Non-Predictive Exploration (NPE): NPE is a non-predictive frontier-based exploration method where information gain is naively estimated by assuming all unobserved areas within a frontier region are traversable. NPE is selected to investigate the influence of map prediction on exploration of unknown environments.

2. AEM Exploration (AEM-E) [14]: AEM-E utilizes the SOTA AEM model for map prediction for frontier-based exploration.

3. T-AEM Exploration (T-AEM-E): T-AEM-E integrates the T-AEM map prediction approach for frontier-based exploration.

4. 4CNet without CN Exploration (4CNet-C-E): *4CNet-C-E* is a predictive exploration method that utilizes *4CNet* for map prediction; however, without the *CN* module to predict confidence scores. This method is used to investigate the impact of confidence scores in the evaluation of expected information gain during frontier selection for *4CNet*.

6) Procedure: A total of 72 trials were conducted in the two environment sizes, using the three energy budgets. Each combination of environment size and energy budget was repeated three times with the different initial robot positions in Fig. 7(b).

7) Results: Table II presents the percentage of area coverage with respect to the three energy budgets for *4CNet-E* and the comparison exploration methods in both environment sizes. In general, *4CNet-E* achieved the highest percentage of area coverage regardless of environment size and energy budget.

NPE achieved a lower coverage percentage than *4CNet-E* as it treated unobserved areas as traversable. This resulted in inaccurate information gain estimations, as unobserved regions contained both free space and obstacles. AEM-E and T-AEM-E both utilized predicted maps for expected information gain estimation, however, their lower coverage compared to *4CNet-E* was mainly due to inaccuracies in their map predictions as a result of the aforementioned single pass prediction and fixed target prediction window.

4CNet-C-E had higher coverage than NPE, AEM-E and T-AEM-E. However, since *4CNet-C-E* assumes all map predictions have equal confidence (uniform confidence score), it had lower area coverage compared to *4CNet-E*. *4CNet-E*, on the other hand, uses the predicted confidence scores to weight pixels in the predicted map; thereby, improving accuracy of expected information gain estimates for each exploration frontier. As a result, *4CNet-E* enabled each robot to prioritize exploration in areas of high uncertainty (low confidence), and directly obtaining observations in these regions, to reduce uncertainty in future map predictions, leading to more information gain through coverage. The Friedman Test showed a statistically significant difference was found for percentage of area coverage across the environment sizes and energy budgets for all exploration methods ($p < 0.001$). Post-hoc Wilcoxon Signed-rank tests with a Bonferroni correction between *4CNet-E* and each exploration method showed that *4CNet-E* had a statistically significant higher area coverage than each of these exploration methods ($p < 0.0125$).

TABLE II
COMPARISON OF EXPLORATION AREA COVERAGE (PERCENTAGE)

Env. Size	15 m × 15 m ($225\ \mathrm{m}^2$)			30 m × 30 m ($900\ \mathrm{m}^2$)		
Method \ Energy	Low	Med	High	Low	Med	High
NPE	18%	30%	39%	18%	31%	42%
AEM-E	18%	30%	40%	18%	30%	42%
T-AEM-E	18%	31%	42%	19%	32%	43%
4CNet-C-E (ours)	21%	36%	49%	18%	34%	48%
4CNet-E (our method)	23%	41%	56%	21%	42%	58%

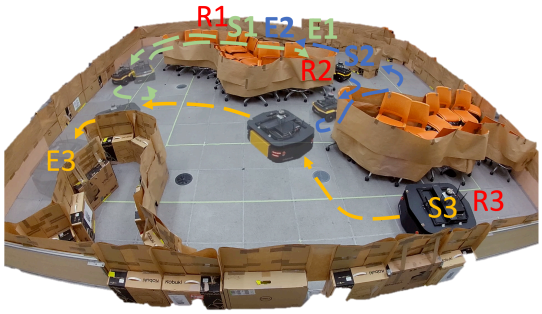
VIII. EXPERIMENTS

We conducted an extensive experiment with *4CNet-E* in an 8.5 m by 8.5 m cluttered physical 3D environment consisting of irregular shaped obstacles, Fig. 8(a)-(b), with 100% CSP. Two Jackal robots and one Ridgeback robot from Clearpath Robotics were deployed with Velodyne 360-degree LiDARs. The ground truth heightmap was obtained with RTAB-Map using a single Jackal robot teleoperated prior to the trial.

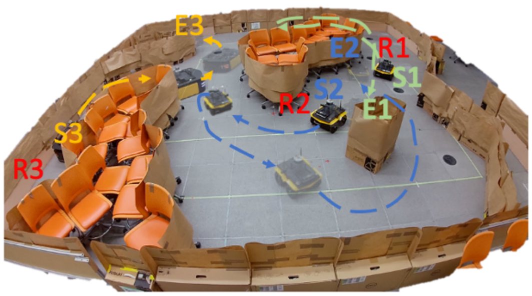
The robots moved with linear and angular velocities of $0.15\ \mathrm{m/s}$ and $0.3\ \mathrm{rad/s}$, respectively. The three robots started at three random start positions, S1, S2 and S3, and ended exploration at end positions E1, E2 and E3, Fig. 8(a)-(b). During the experiment, the two Jackal robots (R1-R2) had an energy budget of 15 m, while the Ridgeback robot (R3) had an energy budget of 8 m. These energy budgets allowed all three robots to achieve up to a maximum of 45% coverage, thereby, requiring map prediction to complete coverage.

A. Exploration with Map Prediction Results

Robot map predictions are shown in Fig. 9(a) – (c) at time steps of 0, 103, 196, and 324s. Each robot's observed regions are represented in white and predicted regions are represented in gray. At each time step, individual robots generated map predictions utilizing direct observations M_i^{obs} and trajectory information δ_t exchanged with nearby robots. In particular, at 103s, R3 and R2 exchanged their trajectories, presented in Figs. 9(b) and (c) by yellow (R3) and blue (R2) dotted lines, respectively. A trajectory exchange also occurred between R3 and R1 at 196s represented by yellow (R3) and green (R1) dotted lines in Fig. 9(b) and (c), respectively. Using both trajectories obtained from R1 and R2, R3 was able to predict the unobserved regions of the obstacle in the middle of the environment in Fig. 9(c). The exploration was completed at 324s, where all three robots depleted their energy budget with an area coverage of 41%, 34% and 35% for R1, R2 and R3, respectively. The obstacle contours of the predicted maps at 324s in Fig. 9(a)-(c), are consistent with the ground truth obstacles, Fig. 9(d), in terms of completeness and alignment.



(a)



(b)

Fig. 8. Resource limited exploration in a real-world unstructured environment with irregularly shaped obstacles and three mobile robots from two viewpoints; (a) for the trajectory of R3, (b) for the trajectory of R2. R1-R3 are Robots 1-3. S1-S3 denote the starting positions of R1-R3. E1-E3 denote the final positions of R1-R3.

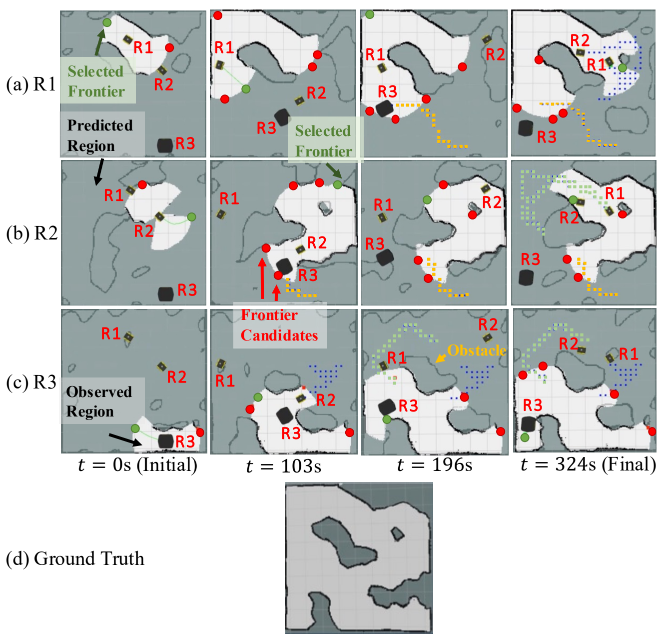


Fig. 9. (a) - (c) presents the map predictions at four distinct time steps for Robots 1-3: 0s (initial), 103s, 196s and 324s (final). Observed regions are in white and target regions for map prediction are in gray. The communicated trajectories between R1, R2 and R3 are represented by green, blue, and yellow dotted lines, respectively. Red circles represent frontier candidates, while green circles represent the selected frontier; and (d) represents the ground truth map.

B. Confidence Prediction Results

The generated confidence maps for the three robots are in Fig. 10. Blue and red regions represent low and high uncertainty in the robot map predictions. Namely, a region with high prediction uncertainty has a low confidence in the robot's map prediction. At 0s, the confidence map for R1 had high uncertainty in the top left region of the environment, Fig. 10(a). As a result, R1 selected a frontier goal in the adjacent area, as indicated by the green Selected Frontier in Fig. 9(a). The confidence map had lower prediction uncertainty in areas with exchanged robot trajectories due to the additional spatial context provided by trajectory embeddings. This was seen at 103s, where the shared trajectories between R3 (yellow dotted line, Fig. 10(b)) and R2 (blue dotted line, Fig. 10(c)) decreased the prediction uncertainties in the corresponding regions as noted by the blue regions. Consequently, at 103s, R2 chose a frontier goal in the upper right area of the environment towards an unexplored region with higher prediction uncertainty, Fig. 9(b). Similar cases of trajectory-influenced confidence and goal selection were observed at 196s and 324s for R1 and R3. A video of our *4CNet-E* approach in both the simulated and real-world unknown environments with irregularly shaped obstacles is presented on our YouTube channel at https://youtu.be/QtviqC-MtEM.

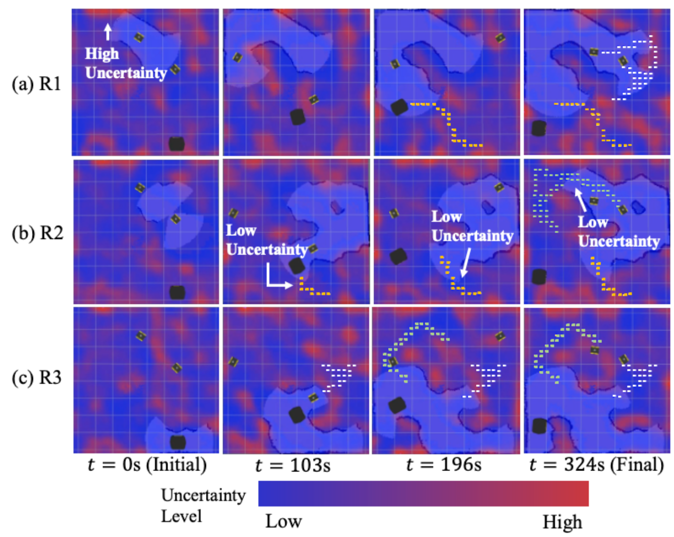


Fig. 10. (a) – (c) presents the confidence maps of Robot 1, 2 and 3 at time steps 0s, 103s, 196s and 324s. Red and blue represent high and low prediction uncertainty, respectively. White, green, and yellow dotted lines represent the communicated R1, R2, and R3 trajectories.

IX. Conclusion

In this paper, we present a novel robot exploration with map prediction architecture called *4CNet-E* that consists of a *Perception and Communication* subsystem, *4CNet* for map prediction, and an *Exploration Planner*. *4CNet* uniquely integrates three components: confidence awareness, contrastive pre-training, and a conditional consistency model for map prediction during resource-limited robot exploration. Our main contributions include: 1) the first utilization of a conditional consistency model in the development of a map prediction network for prediction of the spatial layouts in partially explored environments; 2) the unique application of contrastive learning for pre-training a trajectory encoder in order to consider both static and dynamic environmental elements; and 3) the introduction of a confidence network for map prediction for guiding robots towards areas of high uncertainty to improve map prediction accuracy within constrained energy budgets.

Through extensive simulated comparison experiments, *4CNet-E* was shown to have better performance in terms of map prediction accuracy, and area coverage when compared to heuristic and learning-based methods. Real-world experiments highlight the ability of *4CNet-E* to provide high quality map predictions containing obstacle contours consistent with ground truth maps. Future work will include optimizing map prediction speed by reducing the number of time steps required to generate accurate map predictions using consistency models. We will also test *4CNet-E* in larger real-world environments with uneven terrains to further validate its performance in diverse and challenging scenarios.

Acknowledgment

The authors would like to thank the following members of the ASBLab at UofT, Yuntao Cai, Haitong Wang, Daniel Choi, and Angus Fung for their invaluable discussions and assistance with the physical experimental set-up.

References

[1] J. P. Queralta *et al.*, "Collaborative Multi-Robot Search and Rescue: Planning, Coordination, Perception, and Active Vision," *IEEE Access*, vol. 8, pp. 191617–191643, 2020.

[2] H. Wang, A. H. Tan, and G. Nejat, "NavFormer: A Transformer Architecture for Robot Target-Driven Navigation in Unknown and Dynamic Environments," *arXiv: 2402.06838*, 2024.

[3] X. Zheng, S. Koenig, D. Kempe, and S. Jain, "Multi-robot forest coverage for unweighted terrain," *IEEE Trans. Robot.*, vol. 26, no. 6, pp. 2083–2086, 2010.

[4] M. J. Schuster *et al.*, "The ARCHES Space-Analogue Demonstration Mission: Towards Heterogeneous Teams of Autonomous Robots for Collaborative Scientific Sampling in Planetary Exploration," *IEEE Robot. Autom. Lett.*, vol. 5, no. 4, pp. 5315–5322, 2020.

[5] G. Sakayori and G. Ishigami, "Energy-aware trajectory planning for planetary rovers," *Adv. Robot.*, vol. 35, no. 21–22, pp. 1302–1316, 2021.

[6] M. Lauri, D. Hsu, and J. Pajarinen, "Partially Observable Markov Decision Processes in Robotics: A Survey," *IEEE Trans. Robot.*, vol. 39, no. 1, pp. 21–40, 2023.

[7] G. A. Hollinger and S. Singh, "Multirobot Coordination With Periodic Connectivity: Theory and Experiments," *IEEE Trans. Robot.*, vol. 28, no. 4, pp. 967–973, 2012, doi: 10.1109/TRO.2012.2190178.

[8] Y. Mei, Y. H. Lu, Y. C. Hu, and C. S. G. Lee, "Deployment strategy for mobile robots with energy and timing constraints," *IEEE Trans. Robot.*, vol. 22, no. 3, pp. 2816–2821, 2006.

[9] J. Gielis, A. Shankar, and A. Prorok, "A Critical Review of Communications in Multi-robot Systems," *Curr. Robot. Reports*, vol. 3, no. 4, pp. 213–225, 2022.

[10] K. Ebadi *et al.*, "Present and Future of SLAM in Extreme Environments: The DARPA SubT Challenge," *IEEE Trans. Robot.*, vol. 40, pp. 936–959, 2023.

[11] W. Burgard, M. Moors, C. Stachniss, and F. Schneider, "Coordinated Multi-Robot Exploration," *IEEE Trans. Robot.*, vol. 21, no. 13, pp. 376–386, 2005.

[12] A. Elhafsi, B. Ivanovic, L. Janson, and M. Pavone, "Map-Predictive Motion Planning in Unknown Environments," *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 8552–8558, 2020.

[13] L. Wang, H. Ye, Q. Wang, Y. Gao, C. Xu, and F. Gao, "Learning-based 3D Occupancy Prediction for Autonomous Navigation in Occluded Environments," *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pp. 4509–4515, 2021.

[14] R. Shrestha, F. P. Tian, W. Feng, P. Tan, and R. Vaughan, "Learned map prediction for enhanced mobile robot exploration," *IEEE Int. Conf. Robot. Autom.*, pp. 1197–1204, 2019.

[15] A. H. Tan and G. Nejat, "Enhancing Robot Task Completion Through Environment and Task Inference: A Survey from the Mobile Robot Perspective," *J. Intell. Robot. Syst. Theory Appl.*, vol. 106, no. 4, 2022.

[16] M. Saroya, G. Best, and G. A. Hollinger, "Online Exploration of Tunnel Networks Leveraging Topological CNN-based World Predictions," *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pp. 6038–6045, 2020.

[17] S. K. Ramakrishnan, Z. Al-Halah, and K. Grauman, "Occupancy Anticipation for Efficient Exploration and Navigation," in *European Conference on Computer Vision*, 2020, p. 2020.

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

[19] M. Itkina, "Perception Beyond Sensors Under Uncertainty," *Proc. Thirty-Fifth AAAI Conf. Artif. Intell.*, pp. 15716–15717, 2021.

[20] T. Roddick and R. Cipolla, "Predicting Semantic Map Representations From Images Using Pyramid Occupancy Networks," *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pp. 11135–11144, 2020.

[21] H. J. Chang, S. Member, C. S. G. Lee, Y. Lu, and Y. C. Hu, "P-SLAM : Simultaneous Localization and Mapping With Environmental-Structure Prediction," *IEEE Trans. Robot.*, vol. 23, no. 2, pp. 281–293, 2007.

[22] D. P. Strom, F. Nenci, and C. Stachniss, "Predictive Exploration Considering Previously Mapped Environments," *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 2761–2766, 2015.

[23] A. J. Smith and G. A. Hollinger, "Distributed inference-based multi-robot exploration," *Auton. Robots*, vol. 42, no. 8, pp. 1651–1668, 2018.

[24] M. Luperto, F. Amadelli, M. Di Berardino, and F. Amigoni, "Mapping beyond what you can see: Predicting the layout of rooms behind closed doors," *Rob. Auton. Syst.*, vol. 159, p. 104282, 2023.

[25] Z. Chen, S. Bai, and L. Liu, "Efficient Map Prediction via Low-Rank Matrix Completion," *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 13953–13959, 2021.

[26] Y. Song, P. Dhariwal, M. Chen, and I. Sutskever, "Consistency Models," *arXiv 2303.01469*, 2023.

[27] P. Dhariwal and A. Nichol, "Diffusion Models Beat GANs on Image Synthesis," *Adv. Neural Inf. Process. Syst.*, vol. 11, no. NeurIPS 2021, pp. 8780–8794, 2021.

[28] S. Y. Hayoun, E. Zwecher, E. Iceland, A. Revivo, S. R. Levy, and A. Barel, "Integrating Deep-Learning-Based Image Completion and Motion Planning to Expedite Indoor Mapping," *Comput. Res. Repos.*, 2020.

[29] Y. Katsumata, A. Kanechika, A. Taniguchi, L. El Hafi, Y. Hagiwara, and T. Taniguchi, "Map completion from partial observation using the global structure of multiple environmental maps," *Adv. Robot.*, vol. 36, pp. 279–290, 2022.

[30] P. Zhong, B. Chen, Y. Cui, H. Song, and Y. Sheng, "Space-Heuristic Navigation and Occupancy Map Prediction for Robot Autonomous Exploration," *Proc. Algorithms Archit. Parallel Process.*, vol. 13155, pp. 578–594, 2022.

[31] Z. Shen, L. Kästner, and J. Lambrecht, "Spatial Imagination With Semantic Cognition for Mobile Robots," *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pp. 2174–2180, 2021.

[32] K. Katyal, K. Popek, C. Paxton, P. Burlina, and G. D. Hager, "Uncertainty-aware occupancy map prediction using generative networks for robot navigation," *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 5453–5459, 2019.

[33] M. Bernardine Dias, R. Zlot, N. Kalra, and A. Stentz, "Market-based multirobot coordination: A survey and analysis," *Proc. IEEE*, vol. 94, no. 7, pp. 1257–1270, 2006.

[34] M. Otte, M. J. Kuhlman, and D. Sofge, "Auctions for multi-robot task allocation in communication limited environments," *Auton. Robots*, vol. 44, no. 3–4, pp. 547–584, 2020.

[35] A. H. Tan, F. P. Bejarano, Y. Zhu, R. Ren, and G. Nejat, "Deep Reinforcement Learning for Decentralized Multi-Robot Exploration With Macro Actions," *IEEE Robot. Autom. Lett.*, vol. 8, no. 1, pp. 272–279, 2023.

[36] Z. Chen, B. Subagdja, and A. H. Tan, "End-to-end Deep Reinforcement Learning for Multi-agent Collaborative Exploration," *Proc. IEEE Int. Conf. Agents,* pp. 99–102, 2019.

[37] D. He, D. Feng, H. Jia, and H. Liu, "Decentralized exploration of a structured environment based on multi-agent deep reinforcement learning," *Proc. Int. Conf. Parallel Distrib. Syst. - ICPADS*, pp. 172–179, 2020.

[38] A. D. Haumann, K. D. Listmann, and V. Willert, "DisCoverage : A new Paradigm for Multi-Robot Exploration," *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 929–934, 2010.

[39] R. G. Colares and L. Chaimowicz, "The next frontier: Combining information gain and distance cost for decentralized multi-robot exploration," *31st ACM/SIGAPP Symp. Appl. Comput.*, pp. 268–274, 2016.

[40] C. Stachniss, Ó. M. Mozos, and W. Burgard, "Speeding-Up Multi-Robot Exploration by Considering Semantic Place Information," *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 1692–1697, 2006.

[41] A. Bautin, O. Simonin, and F. Charpillet, "MinPos : A novel frontier allocation algorithm for multi-robot exploration," *Proc. 5th Int. Conf. Intell. Robot. Appl.*, pp. 496–508, 2012.

[42] R. Zlot, A. T. Stentz, M. B. Dias, and S. Thayer, "Multi-Robot Exploration Controlled by a Market Economy," *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 3, pp. 3016–3023, 2002.

[43] Z. Yan, N. Jouandeau, and A. A. Cherif, "Multi-robot decentralized exploration using a trade-based approach," *8th Int. Conf. Informatics Control. Autom. Robot.*, vol. 2, pp. 99–105, 2011.

[44] L. T. Nguyen, J. Kim, and B. Shim, "Low-Rank Matrix Completion: A Contemporary Survey," *IEEE Access*, vol. 7, pp. 94215–94237, 2019.

[45] B. Yamauchi, "A frontier-based approach for autonomous exploration," *Proc. IEEE Int. Symp. Comput. Intell. Robot. Autom.*, pp. 146–151, 1997.

[46] N. Christofides, "Worst-Case Analysis of a New Heuristic for the Travelling Salesman Problem," *Oper. Res. Forum*, vol. 3, no. 1, pp. 1–4, 2022.

[47] M. Glover, Fred; Laguna, *Tabu Search*. 1998.

[48] H. Azpúrua, G. M. Freitas, D. G. Macharet, and M. F. M. Campos, "Multi-robot coverage path planning using hexagonal segmentation for geophysical surveys," *Robotica*, vol. 36, no. 8, pp. 1144–1166, 2018.

[49] Z. Ma, L. Liu, and G. S. Sukhatme, "An adaptive k-opt method for solving traveling salesman problem," *Proc. IEEE 55th Conf. Decis. Control. CDC 2016*, pp. 6537–6543, 2016.

[50] I. Goodfellow *et al.*, "Generative adversarial networks," *Commun. ACM*, vol. 63, no. 11, pp. 139–144, 2014.

[51] S. Salan, E. Drumwright, and K. I. Lin, "Minimum-energy robotic exploration: A formulation and an approach," *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 45, no. 1, pp. 175–182, 2015.

[52] A. Kebaili, J. Lapuyade-Lahorgue, and S. Ruan, "Deep Learning Approaches for Data Augmentation in Medical Imaging: A Review," *J. Imaging*, vol. 9, no. 4, 2023.

[53] T. S. Lembono, E. Pignat, J. Jankowski, and S. Calinon, "Learning Constrained Distributions of Robot Configurations with Generative Adversarial Network," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 4233–4240, 2021.

[54] Y. Liang, B. Chen, and S. Song, "SSCNav: Confidence-Aware Semantic Scene Completion for Visual Semantic Navigation," *Proc. Int. Conf. Robot. Autom.*, pp. 13194–13200, 2021.

[55] M. Labbé and F. Michaud, "RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation," *J. F. Robot.*, vol. 36, no. 2, pp. 416–446, 2019.

[56] J. He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; Sun, "Deep Residual Learning for Image Recognition," *IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 770–778, 2016.

[57] A. Fung, B. Benhabib, and G. Nejat, "Robots Autonomously Detecting People: A Multimodal Deep Contrastive Learning Method Robust to Intraclass Variations," *IEEE Robot. Autom. Lett.*, vol. 8, no. 6, pp. 3550–3557, 2023.

[58] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[59] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The Unreasonable Effectiveness of Deep Features as a Perceptual Metric," *Comput. Vis. Found. CVPR, 2018*, no. 1, p. 13, 2018.

[60] R. Kanopoulos, Nick; Vasanthavada, Nagesh; Baker, "Design of an Image Edge Detection Filter Using the Sobel Operator," *IEEE J. Solid State Circuits*, vol. 23, no. 2, pp. 358–367, 1988.

[61] W. Wang and W. B. Goh, "Multi-robot path planning with the spatio-temporal A* algorithm and its variants," *Proc. Int. Conf. Auton. Agents Multiagent Syst.*, vol. 7068, pp. 313–329, 2011.

[62] C. Rösmann, M. Oeljeklaus, F. Hoffmann, and T. Bertram, "Online trajectory prediction and planning for social robot navigation," *IEEE Int. Conf. Adv. Intell. Mechatronics, AIM*, pp. 1255–1260, 2017.

[63] R. Vitacion and L. Liu, "Procedural Generation of 3D Planetary-Scale Terrains," *Proc. IEEE Int. Conf. Sp. Mission Challenges Inf. Technol.*, pp. 70–77, 2019.

[64] D. Foead, A. Ghifari, M. B. Kusuma, N. Hanafiah, and E. Gunawan, "A Systematic Literature Review of A*Pathfinding," *Procedia Comput. Sci.*, vol. 179, no. 2020, pp. 507–514, 2021.

[65] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, pp. 1929–1958, 2014.

[66] L. Zhang, L. Zhang, X. Mou, and D. Zhang, "FSIM: A feature similarity index for image quality assessment," *IEEE Trans. Image Process.*, vol. 20, no. 8, pp. 2378–2386, 2011.