2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Fast Exploration Planning with Learning-Based Motion Time Prediction for Aerial Robots

Ziyu Wang†, Qianli Dong†, Xuebo Zhang*, Shiyong Zhang, Haobo Xi, Zhe Ma, Mingxing Yuan

Abstract— **Unmanned aerial vehicles (UAVs) have been widely employed to achieve autonomous exploration of 3D unknown environments. However, most existing algorithms suffer from low exploration efficiency caused by inaccurate motion time cost evaluation, which typically leads to the motion inconsistency during the UAV flight. In this work, we propose a learning-based motion time prediction method for real-time evaluating the accurate motion time costs to candidate viewpoints. Specifically, the prediction method takes the current state of the UAV and its surrounding environment features as input to predict the arrival time to each viewpoint. Based on the motion time cost prediction, the UAV can minimize the time wasted by unnecessary acceleration and deceleration during exploration. To further improve the efficiency, we also develop an optimal exploration target decision algorithm that benefits from the predicted motion time costs and the adaptive upper-bound constraints. Simulation and real-world experiments demonstrate that our method can significantly improve the exploration efficiency and increase the average flight speed of the UAV.**

I. INTRODUCTION

Autonomous exploration using unmanned aerial vehicles (UAVs) has found widespread applications in critical scenarios including search and rescue [1], [2], environmental monitoring [3], and geological surveying [4]. However, achieving fast and efficient autonomous exploration in large-scale 3D environments remains a key challenge.

To improve the UAV exploration efficiency, many approaches have been proposed in recent years. Frontier-based methods [5] detect boundaries between explored and unknown regions, guiding robots to these areas for exploration. Early methods employ a greedy strategy, selecting the nearest frontier with the maximum information gain or minimal velocity variation as the next exploration target [6], [7]. To address suboptimal decisions in single-step exploration planning, works in [8], [9] introduced the Traveling Salesman Problem (TSP) to minimize the total exploration path length by considering multiple candidate targets. Alternatively, sampling-based methods [10]-[12] conduct exploration by randomly generating candidate viewpoints, and selecting the optimal next targets based on path costs and exploration gains. Hybrid strategies [13]-[15] combining frontier-based global exploration with sampling-based local exploration have also been developed to further improve the efficiency. However, existing approaches mainly focus on the design of exploration strategy while underestimating the importance of motion time cost evaluation for reaching each candidate viewpoint during the decision process. Specifically, motion cost evaluations often rely on coarse metrics such as path length and velocity differences, which fail to adequately incorporate the UAV's current state and environmental constraints. Such inaccurate evaluation can result in inefficient trajectories and inconsistent target selection, ultimately reducing exploration efficiency. While motion primitive-based methods [16] offer relatively accurate time cost evaluation, they require extensive sampling of motion primitives, leading to substantial computational overhead. Therefore, developing an efficient and accurate method for assessing the motion time cost to reach each candidate viewpoint is essential for improving the exploration efficiency.

To address these challenges, we propose a learning-based motion time cost prediction method that evaluates the cost for each candidate viewpoint in the UAV's vicinity. By leveraging the UAV's current state and features of the surrounding environment, our method can accurately predict the motion time cost in real time. Based on the predicted costs, we develop an optimal exploration target decision strategy using Dijkstra's algorithm. The search is conducted under adaptive upper-bound constraints computed through a lazy evaluation strategy. We compare the proposed method with state-of-the-art baselines in simulation, and the results demonstrate that our method achieves the highest exploration efficiency. Furthermore, we deploy our method on a self-built UAV and validate it in real-world experiment.

The contributions of this work are summarized as follows:

- We propose a learning-based motion time cost prediction method that can accurately evaluate the cost for each candidate viewpoint in real time.
- We propose an optimal target decision algorithm for exploration that leverages the predicted motion time costs and adaptive upper-bound constraints computed during Dijkstra's search.
- We perform extensive simulations and real-world experiments to evaluate our method's efficiency.

II. RELATED WORKS

A. Motion Cost Prediction

Accurate motion time cost prediction is critical for minimizing energy and time waste caused by frequent acceleration and deceleration during autonomous UAV exploration.

This work was supported in part by National Key Research and Development Project under Grant 2022YFB4701800, in part by the China Postdoctoral Science Foundation under Grant Number 2024M751526, in part by the Beijing-Tianjin-Hebei Fundamental Research Cooperation Project under Grant Number 24JCZXJC00390. The authors are with the College of Artificial Intelligence, Nankai University, Tianjin, China. {wziyu, qianlidong}@mail.nankai.edu.cn, zhangxuebo@nankai.edu.cn.

† These authors contributed equally to this work.

∗ Corresponding Author

However, existing methods generally adopt simplified models for cost evaluation. The works in [10], [15], [17]-[19] evaluate motion costs by computing the exploration path length. This simplification often triggers frequent target switching, resulting in exploration inconsistency. Recent studies have attempted to incorporate kinematic constraints. Cieslewski et al. [20] develops a velocity-change-based cost model, and Zhou et al. [8] proposes a motion consistency cost that penalizes angular deviation between the current velocity and the exploration direction. Luo et al. [21] adopts a vertical decomposition method to model acceleration profiles. However, these approaches still have limitations. First, they heavily rely on obstacle-free assumptions and perform motion planning without adequately handling collision avoidance in complex environments. Second, insufficient constraints on acceleration continuity prevent them from satisfying the dynamics of the UAV. The work in [16] proposes a refined evaluation method based on motion primitive sampling, which can achieve accurate time cost evaluation by simultaneously considering velocity, acceleration, and obstacle constraints. Nevertheless, this method require significant computational overhead in dense obstacle environments due to the exhaustive computation of motion primitives. To address these limitations, we design a learning-based motion time cost prediction method to accurately and efficiently evaluate the time cost of reaching each candidate viewpoint from the UAV's current state.

B. Exploration Target Decision

The exploration target decision strategy is also a critical factor influencing the efficiency of autonomous exploration. Existing methods can be primarily categorized into frontier-based, sampling-based and hybrid approaches.

Frontier-based methods leverage the boundaries between explored and unknown spaces to guide exploration. Yamauchi [5] adopts greedy strategies that directly select the nearest frontier as the target. While straightforward, such single-step decision mechanisms lack global optimization capability. To mitigate this, the works in [8], [9] reformulate multi-target exploration as a TSP, aiming to optimize global paths through minimal-cost frontier visitation sequences. However, the NP-hard characteristic of TSP leads to prohibitive computational complexity in large-scale environments.

Sampling-based approaches randomly generate viewpoints in free space, expand Rapidly-exploring Random Trees, and evaluate the information gain of each node to select exploration targets [10]-[12]. A representative work is the Receding Horizon Next-Best-View Planner (RH-NBVP) [10], which iteratively selects paths with maximum information gain from sampled candidate viewpoints. Although it is effective, the simple framework struggles with sampling failure in cluttered spaces due to limited viewpoint diversity.

To overcome the limitations of frontier-based and sampling-based approaches, hybrid frameworks have been proposed. Recent works in [13], [14] integrate frontier detection and Probabilistic Roadmaps to enhance adaptability:

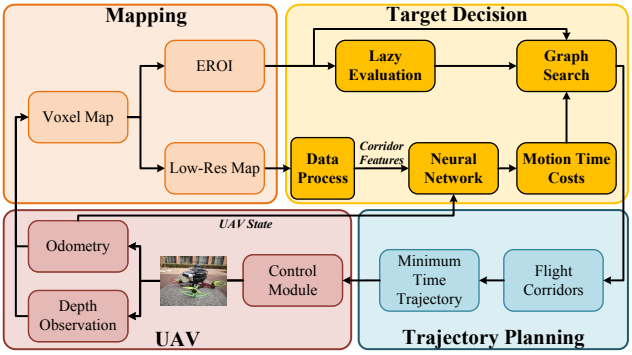


Fig. 1. An overview of the proposed autonomous exploration framework.

when local sampling fails, the roadmap guides the UAV toward nearby global candidate regions. Although these methods mitigate deadlock, their reliance on greedy search may still lead to suboptimal decisions. Therefore, zhang et.al [15] introduced a lazy target evaluation mechanism that adaptively adjusts the upper bound of the target searching algorithm, enabling efficiently optimal target search with reduced computational overhead. Based on the above insights, we propose a exploration target decision strategy that utilizing the predicted motion time costs and the adaptive upper bound of the target searching. This method reduces the computational burden of TSP-based and sampling-based approaches while ensuring globally consistent exploration decisions.

III. METHODOLOGY

A. System Overview

The system architecture of the proposed autonomous exploration framework is illustrated in Fig. 1. First, the mapping module fuses the odometry estimation and depth measurement from onboard sensors to build a voxel map. Based on the voxel map, it simultaneously generates a low-resolution map (LRM) for efficient collision-free path search and identifies explorable regions of interest (EROI) [19] for target decision. Subsequently, a data processing module is applied to the LRM to extract corridor features from all reachable low-resolution voxels (LRVs) within a fixed-depth neighborhood centered at the UAV's current LRV. Then, the UAV's current state and the extracted corridor features are fed into a deep neural network to accurately predict motion time costs from the UAV to each surrounding LRV. Based on these predicted costs, the Dijkstra algorithm is employed to find paths to candidate viewpoints. During the search, a lazy evaluation strategy adaptively computes the upper bound of the search range, accelerating the target decision process while ensuring optimal target is found. Based on the path to the selected exploration target, we construct flight corridors to ensure collision-free trajectory generation. Finally, we optimize a safe, smooth, and dynamically feasible minimum-time MINCO [22] trajectory and send the control commands to the UAV for execution.

B. Learning-Based Motion Time Cost Prediction

The learning-based framework consists of two main modules: a data processing module for extracting corridor features and a neural network module for predicting the motion time costs.

1) Data Process: The LRM is denoted by $\mathcal{M} \subset \mathbb{R}^3$ and consists of LRVs represented by ν. The center position of each LRV ν_i is $\mathbf{o}_i \in \mathbb{R}^3$. We then define a fixed-depth local submap $\mathcal{M}_d \subset \mathcal{M}$ centered at ν_{uav}, the LRV that contains the UAV's current position. The state of the UAV is represented as $\mathbf{S}_{\text{uav}} = [\mathbf{p}_{\text{uav}}^{\mathrm{T}}, \mathbf{v}_{\text{uav}}^{\mathrm{T}}, \mathbf{a}_{\text{uav}}^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^9$, where $\mathbf{p}_{\text{uav}} \in \mathbb{R}^3$, $\mathbf{v}_{\text{uav}} \in \mathbb{R}^3$, and $\mathbf{a}_{\text{uav}} \in \mathbb{R}^3$ denote its position, velocity, and acceleration, respectively.

We adopt a two-step path search method based on Breadth-First Search (BFS) and Depth-First Search (DFS) to efficiently extract the corridor features from the paths connecting ν_{uav} to each reachable LRV $\nu_i \in \mathcal{M}_d$, where the number of reachable LRVs is denoted by N, as detailed in Alg. 1. First, we perform BFS starting from ν_{uav} up to a maximum search depth D. During the search, each visited LRV ν_i stores its parent LRV and the expansion direction to each child LRVs. These records enable reconstruction of the shortest path from ν_{uav} to ν_i (see Fig. 2 (a)). We attach the resulting path information to the local LRM $\mathcal{M}_d$, and denote the augmented map by $\mathcal{M}_d^s$ (Algorithm 1, line 1). For each reachable LRV ν_i, the path from ν_{uav} to ν_i is denoted by P_i, and we defined k_i as the path length.

Subsequently, DFS is applied to $\mathcal{M}_d^s$, starting from ν_{uav}. It iteratively expands each path by following the recorded expansion directions to child LRVs, while dynamically computing corridor between each pair of adjacent LRVs along the path (see Fig. 2 (b), and Algorithm 1, lines 6-16). By maintaining a stack-based corridor collection, the algorithm automatically removes corridor features associated with traversed child LRVs during backtracking to the parent LRV (Algorithm 1, lines 14-19). This mechanism ensures the avoidance of redundant computation of corridor for shared path segments. The corridor features for all reachable LRVs ν_i are represented as $\mathcal{F}_{\text{all}} = \{\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_N\}$, and $\mathbf{F}_i = [\mathbf{f}_{i,1}^{\mathrm{T}}, \mathbf{f}_{i,2}^{\mathrm{T}}, \ldots, \mathbf{f}_{i,D}^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{D\times 6}$ denotes the corridor features from ν_{uav} to ν_i. Here, $\mathbf{f}_{i,j}$ represents the corridor connecting ν_{j-1} and ν_j in P_i, which is shaped as a cuboid. Each corridor is defined as $\mathbf{f}_{i,j} = [l_{i,j}, w_{i,j}, h_{i,j}, \mathbf{p}_{i,j}^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^6$, where $l_{i,j}$, $w_{i,j}$, and $h_{i,j}$ represent the length, width, and height of the corridor, respectively, and $\mathbf{p}_{i,j} \in \mathbb{R}^3$ denotes the corridor center. If the path length k_i is less than D, each remaining entry from $\mathbf{f}_{i,k_i+1}$ to $\mathbf{f}_{i,D}$ is set to the padding vector $\mathbf{x} \in \mathbb{R}^6$ whose elements are all -1, ensuring that $\mathbf{F}_i$ has a fixed dimensionality of $\mathbb{R}^{D\times 6}$ (Algorithm 1, lines 12).

2) Network Architecture: As shown in Fig. 3, the core of the motion time cost prediction framework is a deep neural network that processes the UAV's current state $\mathbf{S}_{\text{uav}}$ and the corridor features $\mathbf{F}_i$ and position $\mathbf{o}_i$ of each reachable LRV ν_i. The network is trained to predict the motion cost for the UAV to reach each reachable LRV, which avoids sampling the UAV's motion primitives and significantly reduces the evaluation time of motion costs while considering the UAV's current state.

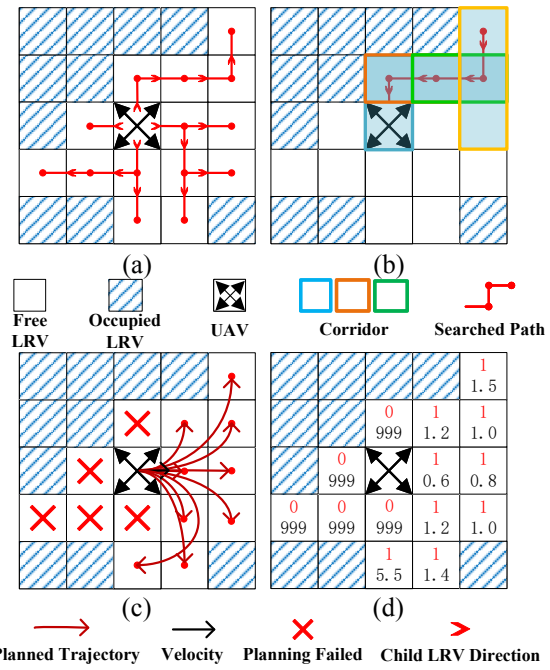


Fig. 2. 2D Illustration of data process. (a) Search for the shortest path to each reachable LRV using BFS. (b) Extract corridor features $\mathbf{F}_i$ for LRV ν_i. (c) Plan trajectories for all reachable LRVs. (d) Ground-truth motion time cost labels are generated using the planned trajectories (shown as black numbers). The red labels indicate whether trajectory planning succeeds (0 for failure, and 1 for success). For any LRV where planning fails, the motion time cost is set to $T_{\text{fail}} = 999s$ and excluded from target decision.

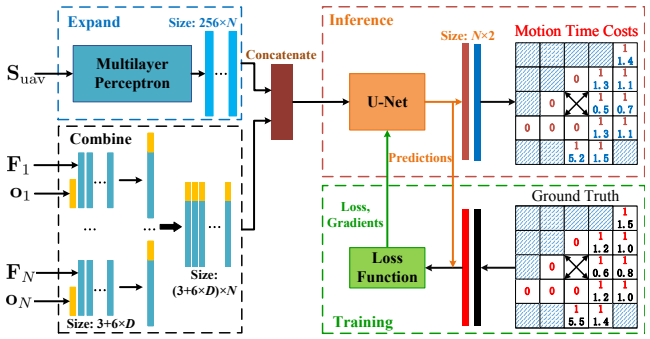


Fig. 3. Network architecture. Our method takes the UAV's current state $\mathbf{S}_{\text{uav}}$, the corridor features $\mathbf{F}_i$ and the position $\mathbf{o}_i$ of each reachable LRV ν_i as input, and predicts the corresponding motion time costs.

First, the UAV state is processed through a multilayer perceptron and concatenated with the corridor features of all reachable LRVs. The fused features are then fed into a U-Net to predict the motion time cost for each ν_i. For the N reachable LRVs, the network produces an output matrix $\mathbf{M}_{\text{p}} \in \mathbb{R}^{N\times 2}$, whose i-th row corresponds to ν_i:

$$\mathbf{M}_\star = \left[\mathbf{M}_\star^\top(\nu_1), \mathbf{M}_\star^\top(\nu_2), \ldots, \mathbf{M}_\star^\top(\nu_N)\right]^\top \in \mathbb{R}^{N\times 2},$$
$$\mathbf{M}_\star(\nu_i) = \left[C_\star(\nu_i), T_\star(\nu_i)\right]^\top \in \mathbb{R}^2, \quad \star \in \{\mathrm{p}, \mathrm{g}\}, \tag{1}$$

where $\mathbf{M}_{\text{p}}(\nu_i)$ denotes the network prediction for ν_i, and

$\mathbf{M}_{\mathrm{g}}(\nu_i)$ denotes the corresponding ground truth. Specifically, $C_{\mathrm{p}}(\nu_i) \in [0,1]$ is the predicted probability that a collision-free and safe trajectory can be planned for ν_i, while $C_{\mathrm{g}}(\nu_i) \in \{0,1\}$ is the ground truth feasibility label (0 for failure, and 1 for success). During inference, if $C_{\mathrm{p}}(\nu_i) < 0.5$, the corresponding LRV ν_i is regarded as infeasible. Moreover, $T_{\mathrm{p}}(\nu_i)$ and $T_{\mathrm{g}}(\nu_i)$ denote the predicted and ground truth motion time cost, respectively.

The ground truth is generated as shown in Fig. 2 (c), (d). Specifically, based on the UAV's state $\mathbf{S}_{\mathrm{uav}}$ and the corridor obtained from Fig. 2(b), a trajectory is planned for each ν_i. If the planning succeeds, the trajectory duration is used as the ground truth motion cost and the corresponding LRV is labeled as 1. Otherwise, the LRV is labeled as 0 and assigned a failure cost T_{fail} (e.g., $T_{\mathrm{fail}} = 999s$). Using the ground truth as supervision, the network is trained with a loss function consisting of two parts: a classification loss and a time prediction loss. The classification loss uses binary cross-entropy (BCE), while the time prediction loss uses mean squared error (MSE) and is computed only for LRVs where a successful trajectory can be planned. The loss is calculated as follows:

$$\begin{aligned}
\mathcal{L} &= \lambda_1 \frac{1}{N}\sum_{i=1}^{N} BCE(\nu_i) + \lambda_2 \frac{1}{N_+}\sum_{i=1}^{N} C_{\mathrm{g}}(\nu_i) MSE(\nu_i), \\
BCE(\nu_i) &= -(C_{\mathrm{g}}(\nu_i)\log(C_{\mathrm{p}}(\nu_i)) \\
&\qquad + (1 - C_{\mathrm{g}}(\nu_i))\log(1 - C_{\mathrm{p}}(\nu_i))), \\
MSE(\nu_i) &= (1 - \frac{T_{\mathrm{p}}(\nu_i)}{T_{\mathrm{g}}(\nu_i)})^2,
\end{aligned} \tag{2}$$

where λ_1 and λ_2 are the weights of the BCE loss and MSE loss, respectively, and $N_+ = \sum_{i=1}^{N} C_{\mathrm{g}}(\nu_i)$ denotes the number of reachable LRVs with successful trajectory planning. To avoid the unequal impact of different magnitudes of $T_{\mathrm{g}}(\nu_i)$ on the loss, we have normalized the MSE function.

The network parameters are optimized with Adam to minimize the loss defined above. Additionally, a learning rate scheduler is adopted to reduce the learning rate when the validation loss does not improve for 10 consecutive epochs.

C. Optimal Exploration Target Decision Algorithm with Adaptive Upper-Bound Constraints

In UAV exploration tasks, we uniformly sample a set of collision-free candidate viewpoints around each EROI as valid viewpoints. Each viewpoint is defined as $\mathbf{c} = [\mathbf{p}_{\mathrm{c}}, \psi]^{\mathrm{T}} \in \mathbb{R}^3 \times SO(2)$, where $\mathbf{p}_{\mathrm{c}}$ is the position of the viewpoint and ψ denotes the yaw orientation. To evaluate the quality of a viewpoint, its utility value $\mathcal{U}(\mathbf{c}_i)$ is calculated as:

$$\mathcal{U}(\mathbf{c}_i) = \mathcal{G}(\mathbf{c}_i) e^{-\lambda \mathcal{T}(\mathbf{c}_i)}, \tag{3}$$

where $\mathcal{G}(\mathbf{c}_i)$ denotes the exploration gain of viewpoint $\mathbf{c}_i$, $\mathcal{T}(\mathbf{c}_i)$ represents the motion time cost from ν_{uav} to the LRV where $\mathbf{c}_i$ is located, and $\lambda > 0$ is a tunable parameter.

To efficiently search the optimal exploration target from the candidate viewpoints, we adopt a method that combines motion time costs with a lazy evaluation strategy. The lazy evaluation strategy proposed by FSMP [15] achieves optimal exploration target by evaluating only a subset of candidate viewpoints, thereby avoiding the high computational burden of calculating exploration gains and motion costs for all candidate viewpoints [15]. Specifically, for the currently evaluated viewpoint $\mathbf{c}_i$, its utility is $\mathcal{U}(\mathbf{c}_i)$. For another candidate viewpoint $\mathbf{c}_j$, if it achieves the maximum possible exploration gain $\mathcal{G}_{\max}$ and requires $\mathcal{U}(\mathbf{c}_j) \geq \mathcal{U}(\mathbf{c}_i)$, its motion cost $\mathcal{T}(\mathbf{c}_j)$ must satisfy the following conditions:

$$\mathcal{T}(\mathbf{c}_j) \leq \mathcal{T}_{\max} = -\frac{1}{\lambda}\ln(\frac{\mathcal{U}(\mathbf{c}_i)}{\mathcal{G}_{\max}}), \tag{4}$$

where $\mathcal{G}_{\max}$ is constrained by the sensor's effective perception range.

However, FSMP evaluates motion cost based on path length, neglecting the UAV's state, which leads to planning inconsistency. To address this limitation, we apply the Dijkstra algorithm to search for the optimal viewpoint, based on the predicted motion time costs and the upper bound of the search, which is evaluated through the lazy evaluation strategy. This design improves the efficiency of deciding the optimal exploration target. It is worth noting that if the network output $\mathbf{M}_{\mathrm{p}}(\nu_i)$ for an LRV ν_i is such that $C_{\mathrm{p}}(\nu_i) < 0.5$, the LRV will not be considered for search.

As shown in Fig. 4, the Dijkstra algorithm searches from boundary LRVs in $\mathcal{M}_d$. The motion time cost to viewpoint $\mathbf{c}_i$ is computed as $\mathcal{T}(\mathbf{c}_i) = \max[T_{\mathrm{yaw}}(\mathbf{c}_i), T(\mathbf{c}_i)]$, determined by the greater value between yaw motion time cost $T_{\mathrm{yaw}}(\mathbf{c}_i)$ and position motion time cost $T(\mathbf{c}_i)$. $T_{\mathrm{yaw}}(\mathbf{c}_i)$ is calculated using a trapezoidal velocity profile determined by the UAV's

Algorithm 1 Data Process for Corridor Features

Input: Low-resolution map $\mathcal{M}$, UAV's current LRV ν_{uav}, maximum search depth D, padding vector $\mathbf{x} \in \mathbb{R}^6$
Output: Corridor features $\mathcal{F}_{\mathrm{all}}$ for all reachable LRVs

$\mathcal{M}_d^s \leftarrow \mathrm{BFS}(\mathcal{M}, \nu_{\mathrm{uav}}, D)$
$\mathcal{F}_{\mathrm{all}} \leftarrow \emptyset$
$\mathcal{S} \leftarrow \emptyset$
DFS$(\nu_{\mathrm{uav}}, \mathcal{S})$
return $\mathcal{F}_{\mathrm{all}}$
function DFS$(\nu_{\mathrm{cur}}, \mathcal{S})$
  if $\nu_{\mathrm{cur}} \neq \nu_{\mathrm{uav}}$ **then**
    $\nu_{\mathrm{par}} \leftarrow \nu_{\mathrm{cur}}$.parent
    $\mathbf{f} \leftarrow \mathrm{ComputeCorridor}(\nu_{\mathrm{par}}, \nu_{\mathrm{cur}})$
    $\mathcal{S}$.push($\mathbf{f}$)
  end if
  $\mathbf{F}_{\mathrm{cur}} \leftarrow \mathrm{PadAndStack}(\mathcal{S}, D, \mathbf{x})$
  $\mathcal{F}_{\mathrm{all}}[\nu_{\mathrm{cur}}] \leftarrow \mathbf{F}_{\mathrm{cur}}$
  for $\nu_{\mathrm{child}} \in \nu_{\mathrm{cur}}$.children **do**
    DFS$(\nu_{\mathrm{child}}, \mathcal{S})$
  end for
  if $\nu_{\mathrm{cur}} \neq \nu_{\mathrm{uav}}$ **then**
    $\mathcal{S}$.pop()
  end if
end function

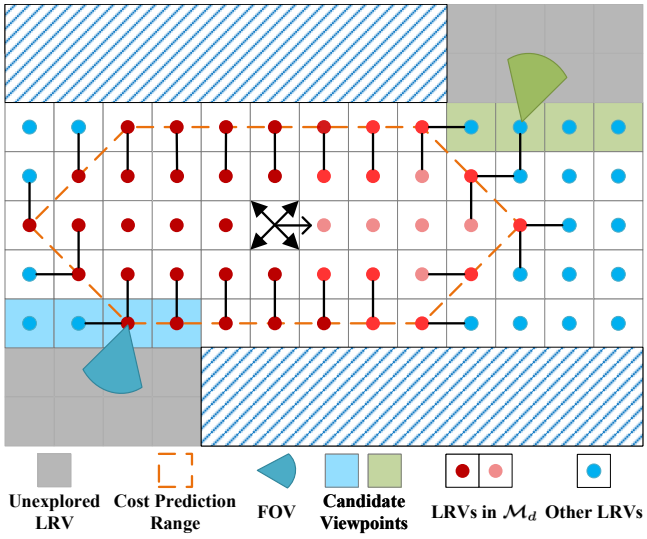


Fig. 4. The 2D illustration of our optimal target decision algorithm. The search starts from the boundary LRVs in $\mathcal{M}_d$, and the red dots represent the motion time cost. The darker the color, the higher the cost.

current yaw and the viewpoint yaw, following the formulation in [19]. $T(\mathbf{c}_i)$ represents the time cost for the UAV from the current LRV ν_{uav} to the LRV of $\mathbf{c}_i$. During the search, the motion time cost $T(\nu)$ for ν is calculated as follows:

$$T(\nu) = \begin{cases} T_{\text{p}}(\nu), & \text{if } \nu \text{ in } \mathcal{M}_d, \\ T_{\text{p}}(\nu_{\text{p}}) + \frac{r}{v_{\max}}, & \text{else}, \end{cases} \tag{5}$$

if $\nu \in \mathcal{M}_d$, its motion time cost is evaluated by the network's prediction as $T_{\text{p}}(\nu)$. Otherwise, the motion time cost is calculated by adding the heuristic time $\frac{r}{v_{\max}}$ to the motion time cost of its parent LRV ν_{p}, where $v_{\max}$ is the UAV's maximum linear speed and r is the resolution of the LRM.

During the search, the Dijkstra algorithm sorts the priority queue based on motion time costs and performs the search within the motion time cost upper bound calculated in (4). When the motion time cost of an LRV popped from the queue exceeds this bound $\mathcal{T}_{\max}$, it indicates that the optimal target has been searched. If the target is in $\mathcal{M}_d$, the corresponding minimum motion time cost path can be obtained from Algorithm 1, line 1 (see Fig. 2 (a)). Otherwise, the path is constructed by combining two segments: (1) the shortest path from ν_{uav} to the starting boundary LRV, also obtained from the BFS in Algorithm 1, and (2) the path searched by the Dijkstra algorithm from the boundary LRV to the optimal exploration target.

After finding the path to the optimal exploration target using Dijkstra's algorithm, we construct safety corridors along it to ensure collision-free flight. Using these corridors as safety constraint, we then optimize a smooth, dynamically feasible MINCO trajectory, which reduces sharp turns and unnecessary speed variations and thereby improves the UAV's tracking performance.

IV. EXPERIMENTS

A. Implementation Details

1) Simulation details: To thoroughly evaluate our method, we conduct simulation and real-world experiments. The simulation framework integrates all algorithms in C++ within a ROS-based computational environment (AMD Ryzen 7 3700X 8-core CPU, NVIDIA GeForce RTX2070 GPU), utilizing Gazebo as the core simulation engine. The simulator UAV platform provided by [23] is equipped with an RGB-D camera with field of view (FoV) parameters of $\theta_h = 115°$ and $\theta_v = 90°$, and a maximum sensing range of 5 m. The maximum velocity of the UAV is limited to 3 m/s, with a maximum acceleration of 4.5 m/s^2. The voxel map resolution is set to 0.1 m, while r is set to 0.7 m.

2) Experiment details: For the real-world experiments, we run the learning-based motion time cost evaluation algorithm on a laptop (Intel Core i7-12700H CPU, NVIDIA GeForce RTX3060 GPU), while the other algorithms are executed on the onboard computer (Intel Core i7-1260P CPU). The UAV is equipped with a Livox Mid 360 LiDAR for localization and mapping, combined with an Intel RealSense D435i depth camera. The D435i has a sensing range of 5 m and FoV parameters of $\theta_h = 87°$ and $\theta_v = 90°$. All other algorithm parameters of the algorithms used in the experiments are kept the same as those in the simulation setup.

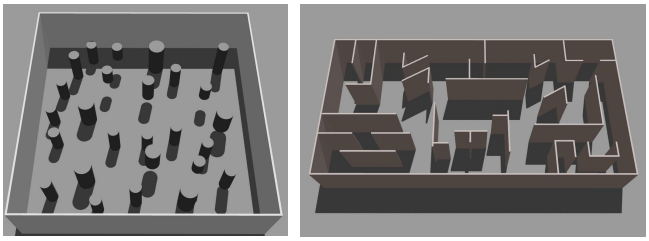
B. Simulation Study

In this section, the proposed method is evaluated in two challenging scenarios, named *Pillars* and *Maze*, respectively, as shown in Fig. 5.

To demonstrate the performance of the proposed method, we compare it with FAEP [9] and FSMP [15] in the simulation trials. Each method is executed 10 times in each scenario. The evaluation metrics are as follows:

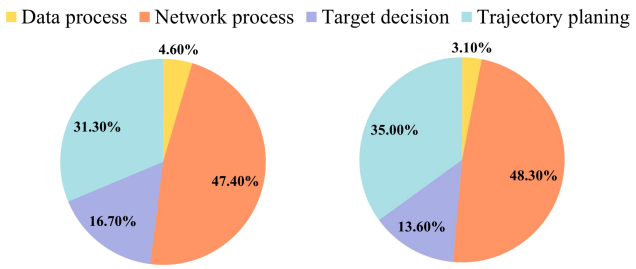
- Exploration Time: The exploration time is defined as the duration until the explored volume exceeds 95% of the total explorable volume of the environment.
- Computation Time: The computation time refers to the average time required for exploration planning.
- Average Speed: The average speed is the mean speed of the UAV throughout the exploration process.

1) Pillars: The size of Pillars scenario is 20×20×5 m^3. We record statistical data after each run and summarize the



(a) Pillars Scenario (b) Maze Scenario

Fig. 5. Two simulation environments.



(a) Time Distribution in Pillars (b) Time Distribution in Maze

Fig. 6. Computational time distribution of our method modules for two simulation scenarios.

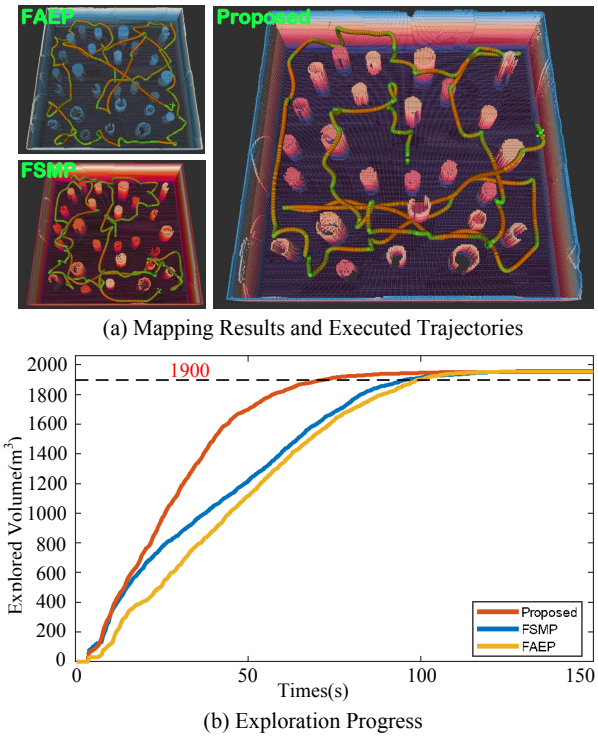
comparison results for each evaluation metric in Table I. As shown in Fig. 6 (a), the computational time distribution of the proposed method during each exploration planning reveals that the time spent on generating the motion cost evaluation and searching for the optimal exploration target constitutes 68.7% of the total execution time. From Fig. 7, it can be observed that our method maintains a consistently high exploration speed despite the increased computation time for motion cost evaluation, resulting in the fastest exploration performance. This improvement can be attributed to our method's integration of the UAV's current state,which effectively reduces the time spent on acceleration and deceleration. As seen in Table I, the proposed method outperforms FAEP and FSMP by 10.2% and 28.0% in average speed, respectively, while also improving exploration efficiency by 27.2% and 24.6%.

2) Maze: The second trial employs the UAV to explore the Maze scenario. The explored area is $40\times20\times3$ m^3, as illustrated in Fig. 5 (b). Similar to the first trial, the time for evaluating the motion cost and searching for the optimal exploration target accounts for 65.0% of the total time (see Fig. 6 (b)). This is because the depth searched in the Alg. 1 is fixed, leading to a constant input size for the network, resulting in consistent computation time. As indicated in Table I and Fig. 8, the proposed method improves the average speed by 31.1% and 33.8% compared to FAEP and FSMP, respectively, while reducing exploration time by 21.7% and 15.5%.

C. Real-world Experiment

To further validate the robustness and effectiveness of our method, we conducted a real-world experiment using a self-built quadrotor UAV in an outdoor environment (see Fig. 9 (a)). As shown in Fig. 9 (b), the outdoor scenario is with cluttered obstacles and its size is $26.5\times19.6\times3$ m^3. Fig. 9 (c), (d) shows the exploration progress, where the UAV successfully explored the majority of the available space within 98 seconds.

In summary, both simulation and experimental results demonstrate that the proposed method achieves efficient exploration. Additional details of all experiments are provided in the supplementary video.



(a) Mapping Results and Executed Trajectories

(b) Exploration Progress

Fig. 7. The exploration progress in Pillars scenario. (a) shows the mapping results and the executed trajectories of FAEP, FSMP, and the proposed method. The color of the trajectory represents the UAV's speed, with green indicating lower speeds and orange indicating higher speeds. (b) shows the exploration progress of three methods.

V. CONCLUSIONS

In this paper, we propose a learning-based motion cost prediction method and a target decision strategy with adaptive upper bounds for UAV exploration in complex 3D environments. Simulation results indicate that our method outperforms state-of-the-art approaches in terms of average speed and exploration efficiency. Furthermore, the real-world experiment confirms that our method is effective for real-world 3D exploration tasks.

One limitation of our method is that localization errors were not considered in the simulated and experimental scenarios. In future work, we plan to incorporate pose drift into our method to enhance its robustness.

REFERENCES

[1] S. Qi, B. Lin, Y. Deng, X. Chen, and Y. Fang, "Minimizing maximum latency of task offloading for multi-UAV-assisted maritime search and rescue," *IEEE Transactions on Vehicular Technology*, vol. 73, no. 9, pp. 13625-13638, 2024.

[2] S. Zhang, X. Zhang, T. Li, J. Yuan, and Y. Fang, "Fast active aerial exploration for traversable path finding of ground robots in unknown environments," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1-13, 2022.

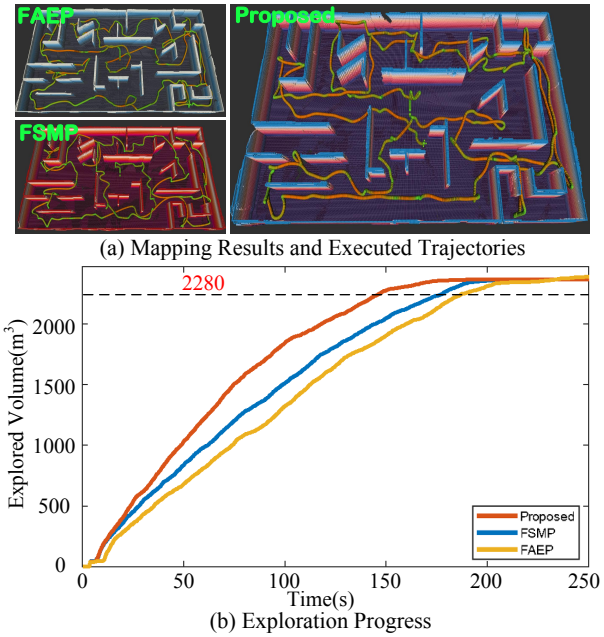
[3] Y. Li, J. Wang, H. Chen, X. Jiang, and Y. Liu, "Object-aware view planning for autonomous 3-D model reconstruction of buildings using a mobile robot," *IEEE Transactions on Instrumentation and Measurement*, vol. 72, pp. 1-15, 2023.

[4] B. Lindqvist, A. Patel, K. Lofgren, and G. Nikolakopoulos, "A tree-based next-best-trajectory method for 3-D UAV exploration," *IEEE Transactions on Robotics*, vol. 40, pp. 3496-3513, 2024.

TABLE I

EXPLORATION TIME, COMPUTATION TIME AND AVERAGE SPEED.

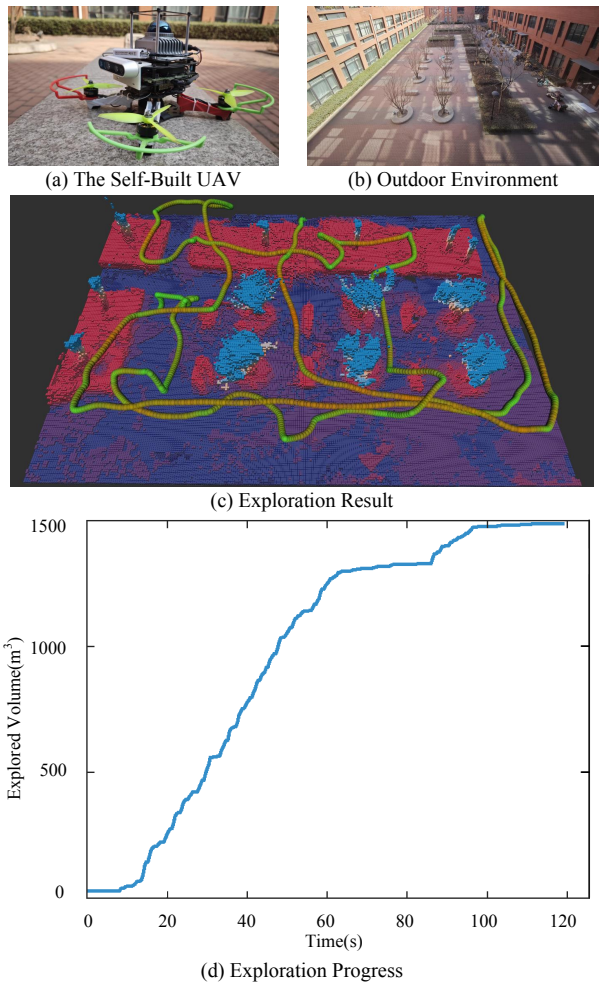
Scenario	Method	Exploration Time (s) Avg	Exploration Time (s) Std	Computation Time (ms) Avg	Computation Time (ms) Std	Average Speed (m/s) Avg	Average Speed (m/s) Std
Pillars	FAEP [9]	98.2	7.0	69.0	82.1	1.66	0.05
	FSMP [15]	94.8	7.9	**13.3**	13.4	1.43	0.04
	Proposed	**71.5**	**3.8**	53.0	**10.6**	**1.83**	**0.03**
Maze	FAEP [9]	194.2	21.5	75.9	73.8	1.48	0.05
	FSMP [15]	179.8	**6.9**	**8.1**	**6.5**	1.45	**0.03**
	Proposed	**152.0**	9.1	50.1	11.7	**1.94**	0.05



(a) Mapping Results and Executed Trajectories

(b) Exploration Progress

Fig. 8. The exploration progress in Maze scenario. (a) shows the mapping results and the executed trajectories of FAEP, FSMP, and the proposed method. (b) shows the exploration progress of three methods.



(a) The Self-Built UAV

(b) Outdoor Environment

(c) Exploration Result

(d) Exploration Progress

Fig. 9. Exploration experiment conducted in an outdoor environment.

[5] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation*, 1997, pp. 146-151.

[6] D. Duberg and P. Jensfelt, "UFOExplorer: Fast and scalable sampling-based exploration with a graph-based planning structure," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2487-2494, 2022.

[7] H. H. González-Banos and J. C. Latombe, "Navigation strategies for exploring indoor environments," *The International Journal of Robotics Research*, vol. 21, no. 10-11, pp. 829-848, 2002.

[8] B. Zhou, Y. Zhang, X. Chen, and S. Shen, "FUEL: Fast UAV exploration using incremental frontier structure and hierarchical planning," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 779-786, 2021.

[9] Y. Zhao, L. Yan, H. Xie, J. Dai, and P. Wei, "Autonomous exploration method for fast unknown environment mapping by using UAV equipped with limited FOV sensor," *IEEE Transactions on Industrial Electronics*, vol. 71, no. 5, pp. 4933-4943, 2024.

[10] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon "next-best-view" planner for 3D exploration," in *2016 IEEE International Conference on Robotics and Automation*, 2016, pp. 1462-1468.

[11] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon path planning for 3D exploration and surface inspection," *Autonomous Robots*, vol. 42, no. 2, pp. 291-306, 2018.

[12] X. Zhang, Y. Chu, Y. Liu, X. Zhang, and Y. Zhuang, "A novel informative autonomous exploration strategy with uniform sampling for quadrotors," *IEEE Transactions on Industrial Electronics*, vol. 69, no. 12, pp. 13131-13140, 2022.

[13] A. Dai, S. Papatheodorou, N. Funk, D. Tzoumanikas, and S. Leutenegger, "Fast frontier-based information-driven autonomous exploration with an MAV," in *2020 IEEE International Conference on Robotics and Automation*, 2020, pp. 9570-9576.

[14] H. Zhu, C. Cao, Y. Xia, S. Scherer, J. Zhang, and W. Wang, "DSVP: Dual-stage viewpoint planner for rapid exploration by dynamic expansion," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2021, pp. 7623-7630.

[15] S. Zhang, X. Zhang, Q. Dong, Z. Wang, H. Xi, and J. Yuan, "FSMP: A frontier-sampling-mixed planner for fast autonomous exploration of complex and large 3-D environments," *IEEE Transactions on*

Instrumentation and Measurement, doi: 10.1109/TIM.2025.3547488.
[16] M. Dharmadhikari, T. Dang, L. Solanka, J. Loje, H. Nguyen, N. Khedekar, and K. Alexis, "Motion primitives-based path planning for fast and agile exploration using aerial robots," in *2020 IEEE International Conference on Robotics and Automation*, 2020, pp. 179-185.
[17] V. M. Respall, D. Devitt, R. Fedorenko, and A. Klimchik, "Fast sampling-based next-best-view exploration algorithm for a MAV," in *2021 IEEE International Conference on Robotics and Automation*, 2021, pp. 89-95.
[18] C. Wang, J. Cheng, W. Chi, T. Yan, and M. Q. H. Meng, "Semantic-aware informative path planning for efficient object search using mobile robot," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 8, pp. 5230-5243, 2021.
[19] Q. Dong, H. Xi, S. Zhang, Q. Bi, T. Li, Z. Wang, and X. Zhang, "Fast and communication-efficient multi-UAV exploration via voronoi partition on dynamic topological graph," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2024, pp. 14063-14070.
[20] T. Cieslewski, E. Kaufmann, and D. Scaramuzza, "Rapid exploration with multi-rotors: A frontier selection method for high speed flight," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017, pp. 2135-2142.
[21] Y. Luo, Z. Zhuang, N. Pan, C. Feng, S. Shen, F. Gao, H. Cheng, and B. Zhou, "Star-Searcher: A complete and efficient aerial system for autonomous target search in complex unknown environments," *IEEE Robotics and Automation Letters*, vol. 9, no. 5, pp. 4329-4336, 2024.
[22] Z. Wang, X. Zhou, C. Xu, and F. Gao, "Geometrically constrained trajectory optimization for multicopters," *IEEE Transactions on Robotics*, vol. 38, no. 5, pp. 3259-3278, 2022.
[23] F. Furrer, M. Burri, M. Achtelik, and R. Siegwart, "RotorS–A modular gazebo MAV simulator framework," *Robot Operating System (ROS)*, pp. 595-625, 2016.