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

Multimodal Fusion-Guided Diffusion Policy for Motion Planning in Rugged and Obstacle-Dense Environments

Haoyu Xi[1,2], Wei Li[1,3*], *Member, IEEE*, Yu Hu[1,3*], *Member, IEEE*

***Abstract*— Motion planning in unstructured environments remains a challenging task, particularly in scenarios with both dense obstacles and rugged terrain, under the requirements for safety and real-time performance. To address these challenges, this paper proposes a multimodal fusion-guided diffusion policy framework, abbreviated as M-DP, which synergistically guided by visual, LiDAR data, and goal targets. An early-fusion mechanism is designed to combine images and LiDAR points, enabling simultaneous semantic and geometric understanding of the environment to guide the diffusion policy in generating obstacle-aware candidate trajectories. Within the diffusion policy module, the Denoising Diffusion Implicit Model (DDIM) is employed to improve real-time performance. Furthermore, a trajectory determination module is proposed that incorporates not only environmental semantics and terrain geometry but also robot dynamic constraints. This ensures that the selected trajectory is dynamically feasible, significantly reducing the risk of tracking failures and enhancing safety in challenging conditions. The selected path effectively balances safety, goal-reaching ability, and bumpiness. Real-world experimental evaluations demonstrate the safety and effectiveness of the framework compared to baseline methods, with ablation studies validating the contributions of key components. Codes and our self-collected dataset are available on https://github.com/xhy1599/M-DP.**

I. INTRODUCTION

With the advancement of robotics, planning in complex environments is of vital importance. Planning algorithms aim to find a feasible path from the start to the destination, ensuring the safety of robots. However, in unstructured environments, finding a clearly safe path is particularly challenging because the free space often lacks connectivity. Unstructured environments, such as mounds, rocks, and vegetation, often feature irregular obstacles and uneven terrain, further fragmenting the traversable area. Dense obstacles and large bumps pose potential dangers in safety-critical applications such as rescues and space exploration. Moreover, it is challenging to ensure safety, including obstacle avoidance and adherence to dynamic constraints, in such complex environments.

Traditional local path planning methods such as Dynamic Window Approach (DWA) [1] and Timed Elastic Band (TEB) [2]–[4] require precise environmental modeling information like costmaps and freespace. In unstructured scenarios, constructing precise environmental models to represent the diverse and dense obstacles may cause considerable computational costs. Therefore, there is a demand to consider the balance of safety and efficiency. In addition, obstacles such as trees and mounds ahead of the robot, create narrow and discontinuous freespace. This may result in the absence of explicit solutions for traditional local path planning methods. With the development of deep learning, learning-based planning methods have attracted increasing attention. Among these, end-to-end planning methods directly process sensor observations and predict future actions, instead of performing perception, prediction, and planning in a hierarchical way, thereby eliminating the strict dependency on costmaps and freespace obtained from preceding perception modules. Compared to hierarchical approaches [5]–[7], end-to-end models can be jointly trained, and the shared backbones increase computational efficiency [8]. However, developing an end-to-end model for safe planning in obstacle-dense unstructured environments remains a challenging task.

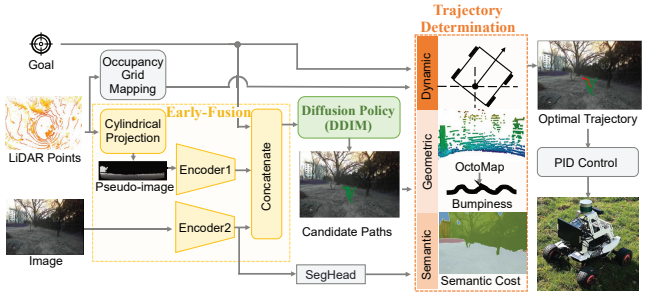

Fig. 1. This paper designs a multimodal fusion-guided diffusion policy framework to ensure the safety of robots in rugged and obstacle-dense environments.

Reinforcement learning (RL) and imitation learning (IL) are two important frameworks inend-to-end planning frameworks. In reinforcement learning, agents interact with the environment in order to learn policies that maximize cumulative rewards. Nevertheless, training robots in the real world may pose safety risks, and training in simulation may face the sim-to-real problem [9]. Imitation learning is a supervised learning approach that learns strategies from expert demonstrations. However, learning the multimodal distribution of expert demonstrations and their sequential correlations remains a challenging task [10].

Diffusion policy [10] has the ability to learn multimodal action distributions and predict action sequences, which combines the idea of the diffusion model and imitation learning. Diffusion model is a probabilistic generative method based on the stochastic differential equation and has the potential

This work was supported by Beijing Natural Science Foundation (L243008), and in part by National Natural Science Foundation of China under Grant No. 62003323 and No. 62176250.

[1]The Research Center for Intelligent Computing Systems, SKLP, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100190, China.
[2]Hangzhou Institute for Advanced Study, University of Chinese Academy of Sciences, Hangzhou, 310024, China.
[3]University of Chinese Academy of Sciences, Beijing, 100049, China.
*Corresponding author. Email: liwei2019@ict.ac.cn, huyu@ict.ac.cn.

to generate higher-quality examples. Diffusion policy is an implicit policy learning framework and uses Denoising Diffusion Probabilistic Model (DDPM) [11] as its foundational framework. However, DDPM has a long sampling time in the inference process. To have a better real-time performance, in this paper, DDPM is used for training, and Denoising Diffusion Implicit Model (DDIM) [12] is deployed for inference because DDIM samples at intervals of multiple steps to offer a faster inference speed.

Meanwhile, current diffusion policy-based planning methods typically rely on a single modality as input. For instance, NoMaD [13] utilizes only visual data while DTG [14] depends on LiDAR. However, in challenging field environments characterized by rugged terrain and dense obstacles, the combination of color and texture information from cameras with accurate ranging information from LiDAR can significantly enhance the planner's ability to identify safe and feasible paths. Early fusion, which means the integration of raw or feature-level data from both sensors at the input stage, is particularly advantageous in such scenarios. Because it enables richer, complementary feature representation and enhances robustness to sensor-specific uncertainties. The Early fusion mechanism allows diffusion policy methods to better reason about complex environments before generating motion trajectories. Nevertheless, feasible early fusion-guided diffusion planning for challenging field environments remains at a preliminary stage, particularly as the influence of robot dynamics on motion execution must still be thoroughly incorporated and evaluated. In rugged and obstacle-dense settings, the absence of dynamic constraints often results in significant tracking errors, actuator saturation, and ultimately, control failure. Embedding dynamic awareness allows the planner to generate trajectories that respect the robot's physical limits, significantly mitigating the risks of rollover, collision, or instability in complex terrains.

This paper proposes a multimodal fusion-guided diffusion policy framework, M-DP, which is collaboratively guided by camera images, LiDAR point clouds, and goals to have a better perception of obstacle information and goal orientation. To reduce the influence of randomness in denoising, M-DP generates multiple trajectories as candidates. Subsequently, a trajectory determination module is proposed, which comprehensively incorporates terrain geometry, surface semantics, goal-oriented and robot dynamic constraints. The main contributions are as follows:

- Propose a motion planning framework for rugged and obstacle-dense environments. The framework introduces a multimodal early-fusion mechanism that integrates goal information, visual and LiDAR sensing at feature level to guide a diffusion policy, significantly improving trajectory reliability through joint semantic and geometric environment understanding.
- Present a dynamics-aware trajectory determination module that incorporates goal-oriented, terrain-aware, and dynamic constraints to select feasible and execution-safe paths, thus effectively reducing the risk of tracking failures and improving navigation safety.
- Deploy the framework on a mobile robot and collect a dataset. Real-world experiments demonstrate that the proposed framework outperforms baseline methods in terms of collision rate and success rate.

II. RELATED WORK

Local planning in unstructured environments. There are some hierarchical methods [15], [16] that learn the semantic information, like traversability or costmaps, and plan trajectories using traditional methods or learning-based methods based on the semantic constraint. But in perception tasks, there is an inherent trade-off between efficiency and accuracy, which affects the safety of the planning module. Recently, end-to-end learning draws a growing popularity [17]. For end-to-end reinforcement learning methods, Josef et al. [18] propose a dense reward signal and train the agent by a Rainbow based architecture [19] for safe planning in unknown rough terrain simulation. But the simulation environments cannot fully represent the sensor noises in the real world, which may cause the sim-to-real problem. SCOML [20] is a offline meta-reinforcement learning method that utilizes the semantics and geometric information to correct the trajectory, which demonstrates the effectiveness in both simulation and real world hybrid terrains. However, it is challenging to design a suitable reward for specific tasks. For imitation learning methods, Pan et al. [21] use online imitation learning algorithm for autonomous driving, which completes racing tasks by learning RGB images from expert data. However, this method may suffer from cumulative errors, posing potential safety risks. César-Tondreau et al. [22] utilize behavioral cloning to learn policies to avoid positive and negative obstacles. But this method uses the discrete action space, which lacks the smoothness of the trajectories. ViPlanner [23] uses geometric and semantic information and employs an imperative learning-based framework to generate path keypoints. Then the path keypoints are refined by a Bi-Level Optimization process. Lee et al. [24] apply RGB-D images and robot states to train a dynamics-aware model, and they design the cost functions to select the sequences that minimize bumpiness and oscillation in orientation. In our work, semantic, geometric, and dynamic information are introduced as the evaluation metrics for trajectory determination, thereby allowing for the collaborative consideration of trajectory safety, goal target, and bumpiness.

Diffusion-based planning methods. With the advancement of the diffusion model [11], [12], [25], several works apply diffusion to enhance the performance of IL, attempting to learn the intrinsic distribution of the demonstrations and handle complex tasks. VINT [26] uses diffusion models to generate intermediate images when performing goal image navigation, while diffusion policy [10] represents visualmotor policy as a conditional denoising process and directly predicts actions from visual observations. NoMaD [13] utilizes VINT as a visual encoder and deploys a single unified diffusion policy for both exploration and objection navigation across diverse environments. Yu et al. [27] add 2D laser data, goal, and global path to guide the diffusion policy

for planning in static and dynamic environments. DP3 [28] encodes sparse point clouds into a compact 3D representation and generates actions conditioned on this representation and the robot poses. DTG [14] takes LiDAR, velocity, and goal as input, and utilizes diffusion model to generate trajectories for global navigation. In this paper, we leverage LiDAR to provide accurate ranging information as a supplement for visual and propose the early-fusion of visual and LiDAR data to enhance obstacle perception.

III. METHODOLOGY

A. The Overall Framework of M-DP

The overall framework of the proposed M-DP is illustrated in Fig. 2. A target goal is selected, and the RGB images and point clouds are acquired by camera and LiDAR, respectively. Then, the early-fusion mechanism of visual and LiDAR is introduced to extract the feature vector, which is shown in the first part of section III.B. The fused feature and the target goal are used to guide the diffusion policy to produce candidate trajectories, which is described in the second part of section III.B. Section III.C explains the trajectory determination module that combines the target goal, semantic, geometric and dynamic information to calculate costs to select the final trajectory.

B. Multimodal Early-Fusion Guided Trajectory Generation

1) ***Early-fusion of Visual and LiDAR Data****:* The early-fusion mechanism is designed to integrate multimodal observations and goal information while maintaining real-time performance. Sensor observations include RGB images obtained from the camera and point cloud data from LiDAR. For the RGB image, ResNet50 [29] is employed to extract the feature, and MLPs are used to compress the feature dimension to obtain the visual feature O_v. The visual network π_v is the combination of ResNet50 and MLPs. Regarding the LiDAR point cloud, reference to [30], we use cylindrical projection to generate pseudo images first. Cylindrical projection leverages the characteristics of LiDAR to convert 3D point clouds into a structured 2D representation. Initially, to reduce the computational costs and filter out useless points, the points beyond a $15m \times 15m$ region are discarded. Subsequently, the 3D point clouds are projected onto a 2D plane. The azimuth angle β_i and polar angle ρ_i represent the angle of the point i in the horizontal and vertical directions. u,v are the corresponding 2D coordinates after the projection.

$$u = \left(\frac{\arctan 2(y_{p,i}/x_{p,i})}{H_{res}} \right) \tag{1}$$

$$v = \left(\frac{\arcsin(z_{p,i}/\sqrt{{x_{p,i}}^2 + {y_{p,i}}^2 + {z_{p,i}}^2})}{V_{res}} \right) \tag{2}$$

where $x_{p,i}$,$y_{p,i}$,$z_{p,i}$ are the coordinates of the point clouds, and H_{res}, V_{res} indicate the resolutions.

This projection maps tens of thousands of point clouds point clouds $\mathcal{P} = \{p_i\}_{i=1}^{S}$ into a pseudo image I_p of size $H \times W \times 3$, capturing the robot's full 360-degree surroundings. The pseudo image is then cropped to retain the 180-degree forward view. To better obtain the initial space attribute of the point clouds, the point cloud network π_p, which includes the PointNet [31] and MLPs, processes the pseudo image to generate the point cloud feature O_p. Finally, the visual feature O_v, point cloud feature O_p, and goal feature O_g are concatenated in the last dimension as the observation feature O. In this paper, we set the lengths of O, O_v, O_p, and O_g as 274, 256, 16, and 2 respectively, and the size of the cropped pseudo image I_p is 16×600, $r_{max} = 15$, $z_{max} = 3$, $z_m in = -1.5$. The process of early-fusion of visual and LiDAR data is shown in Algorithm 1.

Algorithm 1 Early-fusion of Visual and LiDAR Data

Input: Point clouds $\mathcal{P} = \{p_i\}_{i=1}^{S}$, Point cloud network π_p, RGB image I_{img}, Visual network π_v , Goal O_g, Range parameters: r_{max}, z_{min}, z_{max}

Output: Observation feature O

```
Initialize Pseudo image I_p ∈ R^{H×W×3}
for each point p_i in P do
    ρ_i = sqrt(x_{p,i}^2 + y_{p,i}^2 + z_{p,i}^2)
    if ρ_i > r_max or z_{p,i} < z_min or z_{p,i} > z_max then
        continue
    end if
    β_i = arctan 2(y_{p,i}, x_{p,i})
    u_i = β_i / H_res
    v_i = arcsin(z_{p,i}/ρ_i) / V_res
    color_i = GetColor(z_{p,i}, z_max, z_min)
    I_{p,i} = [v_i, u_i, color_i]
end for
I_p = Crop(I_p)
O_p = π_p(I_p)
O_v = π_v(I_img)
O = Concatenate(O_p, O_v, O_g)
```

2) ***Trajectory Generation Based on Diffusion Policy****:* The trajectory generation module takes the early-fusion feature vector O as input and employs the diffusion policy to predict trajectories. To reduce the impact of randomness in trajectory generation on robot safety, we replicate the initial noise and feed it into the noise prediction network at once to generate num_{traj} candidate trajectories. We define a predicted trajectory a that contains a sequence of n intermediate waypoints, the i-th waypoint is $w_i = (x_{w,i}, y_{w,i})$, where $x_{w,i}$ and $y_{w,i}$ are the waypoint positions in x-axis and y-axis. In this work, considering the time step t, the trajectory generation leverages the observation feature sequences $\mathcal{O}_t = \{O_j\}_{j=t-T_o+1}^{t}$ to generate action sequences $\mathcal{A}_t = \{a_j\}_{j=t}^{T_a-1}$, where T_o and T_a are the observation horizon and the action horizon, O_j and a_j are the observation feature and trajectory at time j. The model is trained using DDPM, which involves two key phases: forward diffusion process and reverse diffusion process.

Forward diffusion process: Randomly select an unmodified action $\mathcal{A}_t^0$, a diffusion iteration k, and a noise ϵ. For each step j, add the ϵ^j into $\mathcal{A}_t^{j-1}$. A noise sequence $\mathcal{A}_t^k$ is sampled from $\mathcal{A}_t^0$ and ϵ^k.

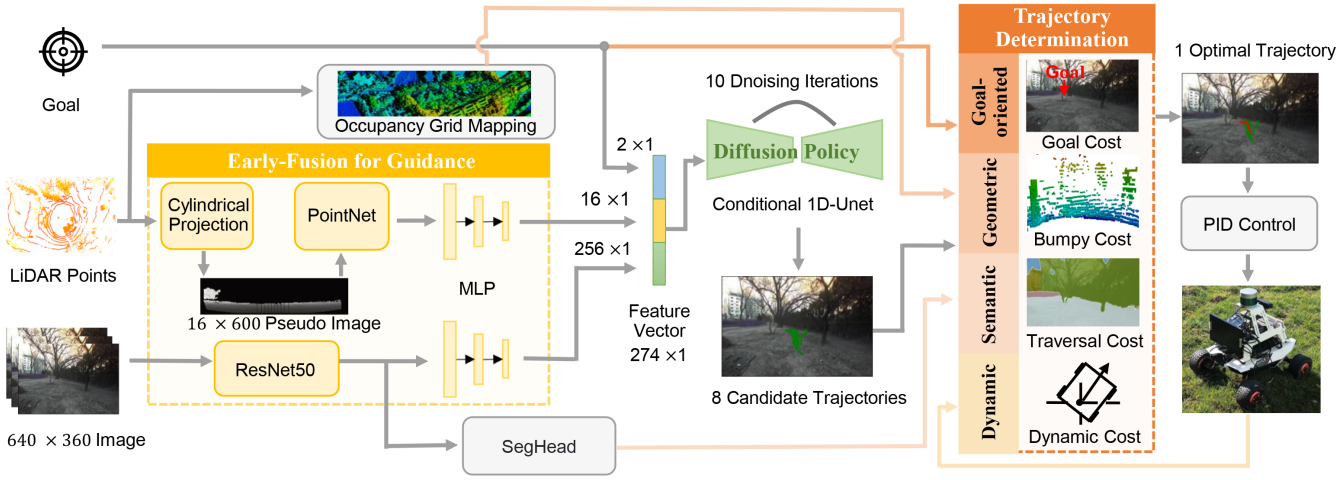

Fig. 2. The M-DP architecture takes three inputs: 3D LiDAR point clouds, RGB images, and a 2D goal target. LiDAR points undergo cylindrical projection to produce a pseudo image processed by PointNet, while RGB images are processed by ResNet50. The outputs from both networks are connected to MLPs for feature extraction. An early-fusion mechanism concatenates these three features to generate the observation feature. The diffusion policy uses this feature to generate candidate trajectories. The trajectory determination module evaluates these candidates using geometric, semantic and dynamic information. Finally, an optimal trajectory is selected and sent to the controller for control command generation.

Reverse diffusion process: Predict the noise in k iterations and generate a series of intermediate action sequences $\{\mathcal{A}_t^k, \mathcal{A}_t^{k-1}, \mathcal{A}_t^{k-2}, \ldots, \mathcal{A}_t^0\}$, until the final noise-free output $\mathcal{A}_t^0$ is obtained. Each denoising step follows the equation:

$$\mathcal{A}_t^{k-1} = \alpha \cdot \left(\mathcal{A}_t^k - \gamma \pi_\theta \left(\mathcal{O}_t, \mathcal{A}_t^k, k\right) + \mathcal{N}\left(\mathbf{0}, \sigma^2 \mathbf{I}\right)\right) \quad (3)$$

where π_θ represents the noise prediction network parameterized by θ, and α, γ are noise schedule parameters. $\mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$ is the zero-mean Gaussian noise with variance σ^2.

We use the Conditional 1D-Unet [32] as the noise prediction network. The noise prediction network π_θ serves as the core component of the diffusion process, tasked with predicting noise conditioned on the observation and action. The loss function for M-DP is formulated using mean squared error (MSE) to quantify the difference between the predicted noise and the ϵ^k. The loss function $\mathcal{L}_{\text{M-DP}}$ is defined as:

$$\mathcal{L}_{\text{M-DP}} = \text{MSE}\left(\epsilon^k, \pi_\theta\left(\mathcal{O}_t, \mathcal{A}_t^0 + \epsilon^k, k\right)\right) \quad (4)$$

During the inference, to have a better inference time, the DDIM is used to replace DDPM, as it can sample at intervals of multiple steps to decrease the diffusion time.

C. Dynamic-Aware Trajectory Determination

The optimal trajectory determination module plays a role in handling multiple trajectories during the inference process. It evaluates each candidate trajectory using semantic, geometric and dynamic constraints as well as goal targeting. The overall cost function J is calculated by the weighted sum of sub-costs. The cost function is designed to ensure the selected trajectory enables the robot to avoid obstacles, achieve the goal, and minimize bumpiness. The individual sub-costs are defined as follows.

TABLE I
TRAVERSAL COST OF DIFFERENT TERRAINS.

Labels	Terrain Description	Cost
Pavement, Road	Smooth and flat surfaces	0
Dirt	Uneven and natural surfaces	4
Grass, Flower, Gravel, Sand	Soft or loose surfaces	10
Tree, Rock, Water, Person, Wall, Fence	Obstacles or impassable areas	500
Sky, Mountain, Building	Background	500

Traversal Cost: The traversal cost J_{trav} utilizes the semantic constraint to enable the robot to avoid obstacles as well as to travel along smooth surfaces as far as possible. J_{trav} is represented as the sum of each waypoint's costs $J_{waypoint}$ along the entire trajectory.

$$J_{trav} = \sum_{i=1}^{n} J_{waypoint} \quad (5)$$

To calculate the traversal cost, Mask2Former [33] is used to obtain terrain information. We use the trained ResNet50 network in early-fusion as the backbone and train the semantic decoder in the COCO dataset [34]. For the traversal cost, different semantic information has different cost values, and the costs of different terrains are shown in Table I.

Goal Cost: Goal Cost J_{goal} is designed to encourage the robot to achieve the goal and calculated by the distance between the end point $w_{end} = (x_{w,end}, y_{w,end})$ of the trajectory and the goal point $w_{goal} = (x_{w,goal}, y_{w,goal})$.

$$J_{goal} = \sqrt{\left(x_{w,goal} - x_{w,end}\right)^2 + \left(y_{w,goal} - y_{w,end}\right)^2} \quad (6)$$

Bumpy Cost: J_{bumpy} is calculated by the variation of z in the z-axis. Different from aerial drones, wheeled robots are constrained to the ground. Consequently, the Z-direction is primarily a reference for planning, rather than being a dimension of planning output. Although the generated

trajectories are in a 2D plane, the OctoMap is used to get the value of the z-axis. The OctoMap [35] is built by LiDAR point clouds and finds the corresponding z-value by the xy-coordinate. For the n intermediate waypoints, the i-th waypoint is $w_i = (x_{w,i}, y_{w,i})$. We define the initial $z_{w,i}$ as the current z value of the robot IMU. Subsequently, find the nearest valid point within the threshold $r_{threshold}$ in OctoMap and use the z-value of the valid point to update $z_{w,i}$. If not found, use the initial $z_{w,i}$ as the z-value. Bumpy cost is defined as:

$$J_{bumpy} = \sum_{i=2}^{n} |z_{w,i} - z_{w,i-1}| \tag{7}$$

where $|\cdot|$ indicates the absolute value.

Dynamic Cost: dynamic cost J_{dy} aims to select trajectories that satisfy the robot's dynamic constraints. In our method, turning radius is introduced to measure dynamic constraints. For the i-th waypoint $w_i = (x_{w,i}, y_{w,i})$ in the trajectory, the orientation β_i is defined as the radians angle between the line from the robot's current position to w_i and the robot's forward direction. The radius_i for waypoints w_i and w_{i+1} is calculated as follows:

$$\text{radius}_i = \frac{\text{Length}(w_{i+1}, w_i)}{2\sin(\Delta\beta_i/2)} \tag{8}$$

where $\text{Length}(\cdot)$ denotes the distance between two waypoints, and $\Delta\beta_i$ is the difference in orientation between w_{i+1} and w_i. The dynamic cost J_{dy} is defined as:

$$J_{dy} = \begin{cases} 0 & \text{if } radius_i \geq radius_{min} \\ 500 & \text{if } radius_i < radius_{min} \end{cases} \tag{9}$$

where $radius_{min}$ is the robot's minimum turning radius.

The overall cost function J is formulated as the weighted sum of all sub-costs.

$$J = W_{bumpy}J_{bumpy} + W_{goal}J_{goal} + W_{dy}J_{dy} + W_{trav}J_{trav} \tag{10}$$

where W_{bumpy}, W_{goal}, W_{dy}, and W_{trav} are the corresponding weights. The overall processes of M-DP training and M-DP inference is shown in Algorithm 2.

Algorithm 2 M-DP process

Input: Parameters: $r_{\max}$, $z_{\min}$, $z_{\max}$, α, γ

M-DP training

Initialize Batch size M, Training iterations N, Dataset $\mathcal{D}$, Noise prediction network π_θ, Point cloud network π_p, Visual network π_v
for iteration $epoch = 1$ to N **do**
  Sample $B = \{(\mathcal{P}_i, I_{img,i}, g_i, a_i)\}_{i=1}^{M} \sim \mathcal{D}$
  O = Early_Fusion ($\mathcal{P}_i$, π_p, $I_{img,i}$, π_v, g_i, $r_{\max}$, $z_{\min}$, $z_{\max}$)
  $\mathcal{O}_t \leftarrow$ Composed of T_o observation horizons O
  Sample a denoising iteration k and a noise ϵ^k, and select a A_t^0
  $\mathcal{A}_t^k = \mathcal{A}_t^0 + \epsilon^k$
  Predict noise $\hat{\epsilon} = \pi_\theta(\mathcal{O}_t, \mathcal{A}_t^k, k)$
  $\mathcal{L} = \|\epsilon_k - \hat{\epsilon}\|^2$
  Update π_p, π_v , π_θ through $\mathcal{L}$
end for

M-DP Inference

Input: Image I_{img}, Point clouds $\mathcal{P}$, Goal target g, Diffusion iterations k

Output: Optimal trajectory $\mathcal{A}_{opt}$

Initialize Empty candidate trajectories $\mathbf{T}$
Sample an initial noise action $\mathcal{A}_t^k$
O = Early_Fusion ($\mathcal{P}$, π_p, I_{img}, π_v, g, $r_{\max}$, $z_{\min}$, $z_{\max}$)
$\mathcal{O}_t \leftarrow$ Composed of T_o observation horizons O
for $candidate = 1$ to num_{traj} **do**
  for $i = k$ to 1 **do**
    $\mathcal{A}_t^{i-1} = \alpha \cdot \left(\mathcal{A}_t^i - \gamma\pi_\theta\left(\mathcal{O}_t, \mathcal{A}_t^i, k\right) + \mathcal{N}\left(\mathbf{0}, \sigma^2\mathbf{I}\right)\right)$
  end for
  Add $\mathcal{A}_t^0$ into $\mathbf{T}$
end for
$Cost_{\mathbf{T}} = CalculateCost(\mathbf{T})$
Select the lowest cost $\mathcal{A}_{opt}$ through $Cost_{\mathbf{T}}$

IV. EXPERIMENTS AND ANALYSIS

A. Experimental Platform

The robot is equipped with a ZED2i stereo camera and an NVIDIA AGX Orin as the onboard computing unit. The chassis is Agilex Scout mini, and the LiDAR is Velodyne VLP-16. Communication is facilitated through the Robot Operating System (ROS) on Ubuntu Linux. The experimental platform is illustrated in Fig. 3.

Fig. 3. Experimental platform. The right picture shows the experimental platform in the experimental scene. The left picture illustrates the specific sensors of the experimental platform.

B. Experimental Environments and Dataset

A real-world dataset is collected in unstructured environments with dense obstacles. The scenes include different obstacles such as trees, rocks, mounds, and different terrain types like grass and dirt paths. During data collection, the robot is operated by a human controller and maintained a constant speed of 0.5 m/s. The ZED2i camera operates at a frequency of 15 Hz, capturing images of size 640×360. The LiDAR operates at a frequency of 15 Hz. The planning frequency is 4 Hz. The total dataset includes 8 scenarios, comprising a total of 101432 pairs of data. Experimental environments and corresponding semantic images are illustrated in Fig. 4. For better model training and testing, we divide the dataset into multiple trajectories, randomly

TABLE II

BASELINE COMPARISONS AND ABLATION STUDY. THE BEST RESULT IS IN BOLD AND SECOND BEST IS UNDERLINE.

Method	Success Rate (%)	Path Length Ratio	Collision Rate (%)	Bumpiness	Cost
Baseline					
DWA [1]	56.8 ± 5.0	1.47 ± 0.10	18.3 ± 3.5	0.38 ± 0.08	78.5 ± 20.3
TEB-U [4]	59.5 ± 4.8	**1.35±0.09**	15.8 ± 2.7	0.36 ± 0.06	69.0 ± 18.6
DP [10]	41.9 ± 5.3	2.03 ± 0.21	46.9 ± 6.8	0.41 ± 0.09	87.7 ± 25.1
NoMaD (image goal) [13]	72.6 ± 2.5	1.44 ± 0.08	17.5 ± 3.2	0.37 ± 0.06	67.4 ± 15.5
NoMaD (no image goal) [13]	63.8 ± 3.5	1.61 ± 0.12	14.2 ± 3.8	0.34 ± 0.07	72.8 ± 17.9
DTG [14]	79.4 ± 1.2	1.41 ± 0.1	5.2 ± 1.4	0.36 ± 0.06	62.8 ± 10.2
Our					
No Image	67.9 ± 3.2	1.42 ± 0.07	5.1 ± 1.2	0.35 ± 0.05	66.3 ± 14.5
No Point Cloud	75.2 ± 3.0	1.43 ± 0.09	6.9 ± 1.5	0.35 ± 0.05	56.7 ± 13.8
M-DP	**82.3±2.1**	1.38 ± 0.06	**3.8±0.9**	**0.29±0.04**	**49.1±11.2**

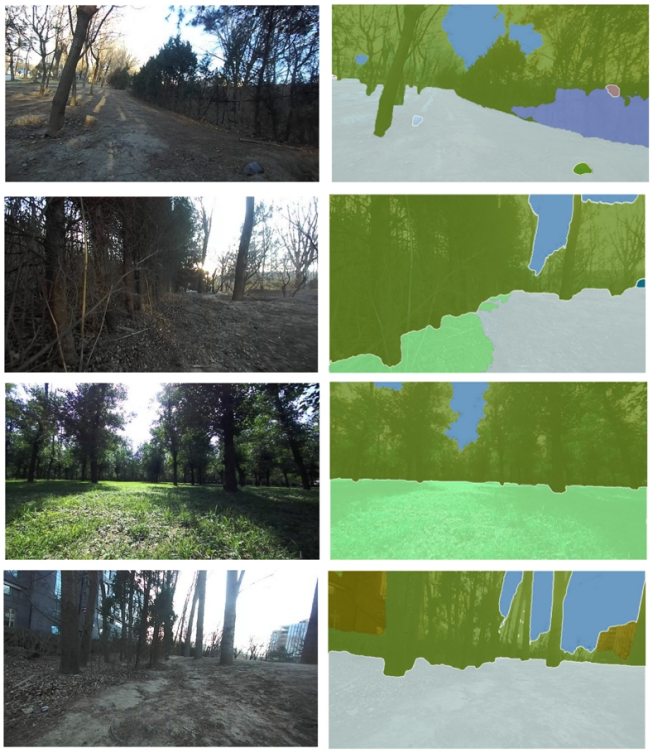

Fig. 4. Experimental environments and corresponding semantic images.

selecting 80% of the dataset for training and 20% for testing. The total number of noise prediction network parameters is 5.68M. Our proposed method is trained on the RTX 3090 GPU for 40 epochs on the same training dataset.

C. Experimental Setup

We conduct a comparative analysis of several approaches: the sampling-based approach DWA [1], the optimization-based approach TEB-U [4], the imitation learning-based approach Diffusion Policy (DP) [10], NoMaD [13], DTG [14]. We compare two types of NoMaD: NoMaD (image goal) and NoMaD (no image goal). The freespace information required by DWA and TEB-U is derived by projecting OctoMap onto a 2D occupancy grid map. Additionally, we conduct ablation studies on our method by evaluating M-DP without visual information (No Image) and M-DP without point cloud information (No Point Cloud). The key parameters for our method are detailed below: diffusion iterations $k = 10$, batch size $= 64$, learning rate $= 10^{-4}$, $radius_{min} = 0.3m$, $W_{bumpy} = 10$, $W_{goal} = 10$, $W_{trav} = 1$, and $W_{dy} = 1$.

To assess the performance of the various approaches, we employ the following evaluation metrics:

Success Rate: An attempt is considered successful if the robot reaches the destination without collisions, getting stuck, or requiring human intervention.

Collision Rate: Rate of collisions in total experiments. The collision rate is additionally calculated to measure the traveling safety in obstacle-dense environments.

Path Length Ratio: The ratio of the traveling trajectory length to the straight-line distance connecting the start and end points, which indicates the path efficiency.

Bumpiness: The average variation in the z-axis.

Cost: The average trajectory cost for each method. The cost metric calculates the mean trajectory cost for successful attempts. Its calculation is introduced in Section III-C.

D. Experiment Result Analysis

Table II presents a comprehensive comparison of various methods in terms of success rate, path length ratio, collision rate, bumpiness, and average cost. Our M-DP method achieves the highest success rate of 82.3%. For path length ratio, M-DP attains 1.38, slightly higher than TEB-U. Although TEB-U shows the shortest path length ratio, its lower success rate of 59.5% suggests that discontinuous free space may impair its polygon fitting performance. In terms of collision rate, M-DP achieves the lowest value at 3.8%, representing a 73.2% reduction compared to NoMaD (no image goal). Notably, even when only using the image data (No Point Cloud), our approach still outperforms baseline methods in both cost and collision rate, validating the effectiveness of the optimal determination module. Bumpiness experiments confirm that M-DP enables stable navigation in unstructured environments, achieving the lowest bumpiness value. Regarding average cost, which calculates the mean trajectory cost for successful attempts, M-DP generates trajectories with the lowest cost, demonstrating the optimal-

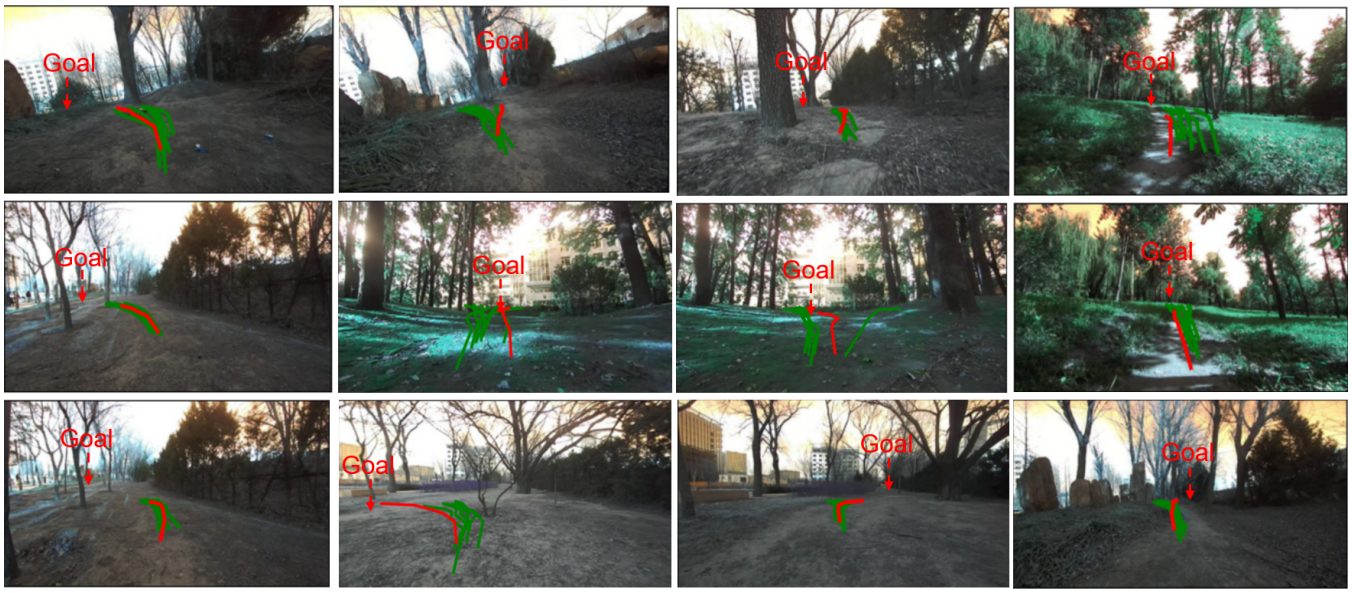

Fig. 5. Trajectory visualization. This visualization displays the trajectories predicted by M-DP in real-world environments. Candidate trajectories are indicated in green, and the optimal trajectory selected by the M-DP is shown in red. The goal is indicated in the picture.

ity of the selected paths. The ablation studies show that our method performs well even with single-modality data. However, when implementing early-fusion, M-DP shows improvements across all metrics. It should be noted that incorporating LiDAR as an input to the diffusion policy leads to a growth in model parameters compared to methods that rely solely on visual input. However, when compared to DTG, a method that also uses LiDAR in its diffusion policy, our approach achieves better performance across multiple metrics while maintaining a substantially smaller model size. Specifically, the DTG_U-Net model requires 1117.89 MB, while our M-DP uses only 131.7 MB. This highlights the practical deployability of our method on real robotic systems.

Table III shows the diffusion loss and action loss on both the training and testing datasets. The diffusion loss is calculated according to Equation (4), while action loss represents the MSE between predicted and ground truth actions. M-DP achieves the lowest diffusion loss and action loss on the test dataset. Although DP shows the lowest diffusion training loss, its generated actions highly deviate from the ground truth, resulting in poor real-world performance. Fig. 5 visualizes trajectories generated and selected by M-DP in different experimental scenarios.

To further verify the real-time performance of DDIM compared to DDPM, we test the average diffusion time with different numbers of diffusion iterations. The experiments are performed on two different hardware platforms: NVIDIA AGX Orin and RTX 3090. AS shown in Table IV, on the AGX Orin platform, DDIM achieves time reductions of 18.1%, 10.0%, 6.6%, and 6.7% for 5, 10, 20, and 100 diffusion iterations, respectively. Similarly, on the RTX 3090 platform, DDIM shows performance improvements of 15.0%, 2.5%, 7.7%, and 6.6% for 5, 10, 20, 100 iterations. The results demonstrate that DDIM achieves faster inference

TABLE III

DIFFUSION LOSS AND ACTION LOSS.

Method	**Diffusion Loss**		**Action Loss**	
	Train	Test	Train	Test
DP [10]	**3.52E-2**	5.87E-2	10.04	13.77
NoMaD (image goal) [13]	4.25E-2	5.74E-2	4.53	5.02
NoMaD (no image goal) [13]	7.63E-2	11.03E-2	3.75	6.9
No Image	4.74E-2	5.73E-2	2.71	4.52
No Point Cloud	4.38E-2	7.77E-2	2.84	4.32
M-DP	4.18E-2	**5.53E-2**	**1.49**	**4.05**

TABLE IV

AVERAGE DIFFUSION TIME (UNIT: MS).

Platform	Method	Diffusion Iterations			
		5	10	20	100
AGX Orin	DDPM	148.4	283.9	533.4	2544.0
	DDIM	121.5	255.4	497.7	2372.6
RTX 3090	DDPM	46.1	72.3	141.3	636.8
	DDIM	39.2	70.5	130.4	597.2

times across all tested iteration counts, which is more suitable for real-time planning. For more real-world robot traveling experiments, please refer to the attached video.

V. CONCLUSIONS

In this paper, we propose a planning framework based on imitation learning that ensures safe robot travel in rugged and obstacle-dense unstructured environments. The trajectory generation module adopts the early-fusion approach of images and LiDAR point clouds to predict candidate trajectories. To balance safety, goal reaching, and bumpiness, a trajectory determination module is designed to combine terrain geometry, surface semantics, and dynamic model of robot to calculate total costs, selecting the optimal trajectory

among multiple candidates. To verify the effectiveness of the proposed method, we collect a dataset in the real world and conduct real-world experiments. Experiments demonstrate the safety and effectiveness over baseline methods. In future work, we plan to use lightweight methods to improve the real-time performance and collect a large-scale dataset for the generalizability across different environments.

REFERENCES

[1] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997.

[2] C. Rösmann, W. Feiten, T. Woesch, F. Hoffmann, and T. Bertram, "Trajectory modification considering dynamic constraints of autonomous robots," in *ROBOTIK 2012; 7th German Conference on Robotics*, 2012, pp. 1–6.

[3] C. Rösmann, F. Hoffmann, and T. Bertram, "Kinodynamic trajectory optimization and control for car-like robots," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 5681–5686.

[4] H. Xi, W. Li, F. Zhao, L. Chen, and Y. Hu, "A safe and efficient timed-elastic-band planner for unstructured environments," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024, pp. 3092–3099.

[5] X. Wang, E. Shang, B. Dai, Y. Nie, and Q. Miao, "Deep reinforcement learning-based off-road path planning via low-dimensional simulation," *IEEE Transactions on Intelligent Vehicles*, pp. 1–12, 2023.

[6] J. Hossain, A.-Z. Faridee, N. Roy, A. Basak, and D. E. Asher, "Covernav: Cover following navigation planning in unstructured outdoor environment with deep reinforcement learning," in *2023 IEEE International Conference on Autonomic Computing and Self-Organizing Systems (ACSOS)*, 2023, pp. 127–132.

[7] X. Meng, N. Hatch, A. Lambert, A. Li, N. Wagener, M. Schmittle, J. Lee, W. Yuan, Z. Q. Chen, S. Deng, G. Okopal, D. Fox, B. Boots, and A. Shaban, "Terrainnet: Visual modeling of complex terrain for high-speed, off-road navigation," in *Robotics: Science and Systems*, 2023.

[8] L. Chen, P. Wu, K. Chitta, B. Jaeger, A. Geiger, and H. Li, "End-to-end autonomous driving: Challenges and frontiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 12, pp. 10 164–10 183, 2024.

[9] L. Brunke, M. Greeff, A. W. Hall, Z. Yuan, S. Zhou, J. Panerati, and A. P. Schoellig, "Safe learning in robotics: From learning-based control to safe reinforcement learning," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 5, no. Volume 5, 2022, pp. 411–444, 2022.

[10] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," in *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[11] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.

[12] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," in *International Conference on Learning Representations*, 2021.

[13] A. Sridhar, D. Shah, C. Glossop, and S. Levine, "Nomad: Goal masked diffusion policies for navigation and exploration," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 63–70.

[14] J. Liang, A. Payandeh, D. Song, X. Xiao, and D. Manocha, "Dtg : Diffusion-based trajectory generation for mapless global navigation," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024, pp. 5340–5347.

[15] A. J. Sathyamoorthy, K. Weerakoon, T. Guan, M. Russell, D. Conover, J. Pusey, and D. Manocha, "Vern: Vegetation-aware robot navigation in dense unstructured outdoor environments," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 11 233–11 240.

[16] X. Cai, M. Everett, J. Fink, and J. P. How, "Risk-aware off-road navigation via a learned speed distribution map," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 2931–2937.

[17] X. Xiao, B. Liu, G. Warnell, and P. Stone, "Motion planning and control for mobile robot navigation using machine learning: a survey," *Autonomous Robots*, vol. 46, 06 2022.

[18] S. Josef and A. Degani, "Deep reinforcement learning for safe local planning of a ground vehicle in unknown rough terrain," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6748–6755, 2020.

[19] M. Hessel, J. Modayil, H. van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. G. Azar, and D. Silver, "Rainbow: Combining improvements in deep reinforcement learning." in *AAAI*, S. A. McIlraith and K. Q. Weinberger, Eds. AAAI Press, 2018, pp. 3215–3222.

[20] A. Yang, W. Li, and Y. Hu, "Scoml: Trajectory planning based on self-correcting meta-reinforcement learning in hybrid terrain for mobile robot," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024, pp. 14 125–14 132.

[21] Y. Pan, C.-A. Cheng, K. Saigol, K. Lee, X. Yan, E. Theodorou, and B. Boots, "Agile autonomous driving using end-to-end deep imitation learning," in *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018.

[22] B. César-Tondreau, G. Warnell, K. Kochersberger, and N. R. Waytowich, "Towards fully autonomous negative obstacle traversal via imitation learning based control," *Robotics*, vol. 11, no. 4, 2022.

[23] P. Roth, J. Nubert, F. Yang, M. Mittal, and M. Hutter, "Viplanner: Visual semantic imperative learning for local navigation," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 5243–5249.

[24] J. H. Lee, J. Choi, S. Ryu, H. Oh, S. Choi, and J. Hwangbo, "Learning vehicle dynamics from cropped image patches for robot navigation in unpaved outdoor terrains," *IEEE Robotics and Automation Letters*, vol. 9, no. 5, pp. 4035–4042, 2024.

[25] J. Carvalho, A. T. Le, M. Baierl, D. Koert, and J. Peters, "Motion planning diffusion: Learning and planning of robot motions with diffusion models," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 1916–1923.

[26] D. Shah, A. Sridhar, N. Dashora, K. Stachowicz, K. Black, N. Hirose, and S. Levine, "Vint: A foundation model for visual navigation," in *Proceedings of The 7th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, J. Tan, M. Toussaint, and K. Darvish, Eds., vol. 229. PMLR, 06–09 Nov 2023, pp. 711–733.

[27] W. Yu, J. Peng, H. Yang, J. Zhang, Y. Duan, J. Ji, and Y. Zhang, "Ldp: A local diffusion planner for efficient robot navigation and collision avoidance," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024, pp. 5466–5472.

[28] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu, "3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations," in *Proceedings of Robotics: Science and Systems (RSS)*, 2024.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 06 2016, pp. 770–778.

[30] J. Liu, G. Wang, Z. Liu, C. Jiang, M. Pollefeys, and H. Wang, "Regformer: An efficient projection-aware transformer network for large-scale point cloud registration," in *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 8417–8426.

[31] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 77–85.

[32] M. Janner, Y. Du, J. B. Tenenbaum, and S. Levine, "Planning with diffusion for flexible behavior synthesis," in *International Conference on Machine Learning*, 2022.

[33] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention mask transformer for universal image segmentation," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 1280–1289.

[34] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 740–755.

[35] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous Robots*, vol. 34, no. 3, pp. 189–206, 2013.