# Real-Time Navigation of Unmanned Ground Vehicles in Complex Terrains With Enhanced Perception and Memory-Guided Strategies

Zhixuan Han, Peng Chen, *Member, IEEE*, Bin Zhou, and Guizhen Yu

***Abstract*—Accurate navigation of unmanned ground vehicles (UGVs) across challenging outdoor terrains demands precise maneuvering amidst diverse surface characteristics, while ensuring collision avoidance. Conventional navigation methodologies often struggle in dynamic environments due to their reliance on pre-mapped data and limitations in real-time multi-sensory data assimilation. To this end, this study proposes a methodology that integrates a diverse array of sensory inputs, including pose data, images, and point clouds, to enhance UGVs' situational awareness and decision-making capabilities. Central to our innovation is the development of a multi-modal reinforcement learning framework, which enhances UGVs' perceptual and decision-making ability. This framework incorporates a lattice-based motion planning algorithm, meticulously calibrated to optimize action selection while respecting UGVs' kinematic constraints. Additionally, a novel dual-training paradigm is introduced, combining curriculum learning and modal separation techniques to address the complexities of multi-modal learning. A notable contribution is the strategic integration of Long Short-Term Memory (LSTM) algorithms, which mitigate information decay and preserve essential navigational strategies over extended periods. The fusion of advanced perception and memory-guided strategies establishes a new standard for autonomous UGV navigation across diverse and unpredictable terrains.**

***Index Terms*—Unmanned ground vehicle, reinforcement learning, unknown environment, complex terrain, navigation.**

## I. Introduction

NAVIGATION, the process of determining a path from one location to another, is fundamental for autonomous systems to operate effectively in diverse and unpredictable terrains. Unmanned ground vehicle (UGV) navigation in complex terrains plays a key role in diverse applications, from emergency rescue and planetary exploration to agricultural automation and environmental monitoring [1], [2].

In conventional navigation methods, simultaneous localization and mapping (SLAM) techniques are key to construct maps of unknown environments. Following this mapping process, the robot's position is precisely determined within these environments using advanced localization algorithms. Critical to this process is the effective collaboration between path planning and control modules, which collectively guide the robot towards its intended destination. However, traditional navigation frameworks, encompassing mapping, localization, and planning modules, confront challenges. Notably, the integration of these modules frequently leads to substantial computational inaccuracies. These inaccuracies tend to accumulate, propagating through various modules and ultimately impairing the performance of these algorithms in real-world applications [3].

Furthermore, these frameworks struggle in dynamic and unstructured environments. They are hampered by limited adaptability to rapidly changing conditions, high computational demands, and a heavy reliance on extensive pre-mapped environmental data. An additional complication arises from the difficulty of integrating real-time sensory feedback into navigation decisions, which is crucial for navigating complex terrains effectively [4].

Deep Reinforcement Learning (DRL) methodologies have catalyzed a paradigm shift in the realm of autonomous navigation, exploiting their formidable representational capacities to assimilate navigation strategies directly from unprocessed sensor data [5]. This progression surmounts the inherent limitations encountered by conventional frameworks within uncharted and unstructured environments, notably their constrained adaptability and the intricacies associated with the integration of instantaneous sensory feedback. Distinguished advancements encompass the multitask learning framework proposed by Natan et al. [6], which capitalizes on RGBD camera data to concurrently execute perception and control operations, alongside the uncertainty-aware, end-to-end trajectory generation technique derived from imitation learning, introduced by Cai et al. [7]. This method is adept at extracting spatiotemporal attributes from front-view camera imagery for enhanced scene comprehension, subsequently facilitating the generation of trajectories devoid of collisions, projected several seconds into the future. These methodologies exemplify the potential of DRL to efficiently map raw sensor data to control strategies, enabling autonomous agents to navigate complex terrains more effectively by reducing computational demands and enhancing adaptability to rapidly changing conditions.

Received 27 May 2024; revised 13 September 2024; accepted 13 November 2024. Date of publication 20 January 2025; date of current version 5 March 2025. This work was supported in part by the National Key Technologies R&D Program of China under Grant 2022YFB4703700 and in part by the National Natural Science Foundation of China under Grant 52272327. The review of this article was coordinated by Dr. Chen Lv. *(Corresponding author: Peng Chen.)*

The authors are with the Key Laboratory of Autonomous Transportation Technology for Special Vehicles, Ministry of Industry and Information Technology, School of Transportation Science and Engineering, Beihang University, Beijing 100191, China (e-mail: buaa_hzx@buaa.edu.cn; cpeng@buaa.edu.cn; binzhou@buaa.edu.cn; yugz@buaa.edu.cn).

Digital Object Identifier 10.1109/TVT.2024.3500002

Despite these advancements, end-to-end DRL approaches encounter limitations. A predominant issue is their reliance on monomodal sensory input—either point clouds [8], [9] or images [10], [11]—which restricts their application in complex, unstructured environments due to the lack of comprehensive perceptual information. Moreover, such methods are prone to the "forgetting phenomenon," where models lose previously learned information over time, as observed in the multi-modal perception-based navigation method by Huang et al. [12], which relies solely on single-frame image inputs. Additionally, the complexity of learning effective strategy networks from multi-modal inputs is exacerbated by scalability issues, necessitating specialized learning schemes to optimize each modality's functionality. Compounding these challenges are the kinematic limitations of vehicles, leading to the generation of ineffective actions or overly simplistic maneuvers that compromise the smoothness and continuity of trajectories.

Addressing the limitations inherent in current deep reinforcement learning approaches for autonomous navigation, this work introduces a comprehensive multi-modal reinforcement learning strategy, uniquely tailored for complex outdoor environments. Our approach synergizes diverse perceptual inputs—such as pose, images, and point clouds—to enhance environmental awareness and decision-making precision for UGVs. The inefficiencies in action selection and information retention are addressed through innovative applications of lattice-based sampling techniques, coupled with the dueling deep Q network (DQN) [13], for refined action space optimization. Furthermore, long short-term memory (LSTM) [14] algorithms are integrated to counteract the forgetting phenomenon, ensuring the retention of critical information over extended periods. The core contributions of this study are summarized as follows:

- *Multi-modal feature fusion and reinforcement learning framework:* A multi-modal feature fusion and reinforcement learning framework is proposed, utilizing an innovative CNN-LSTM network architecture to analyze pose data, depth images, and point cloud data. This architecture employs CNNs to fuse pose data, depth images, and point clouds, and incorporates LSTM for handling pose and depth images, addressing the challenge of rapid information loss in dynamic environments. Additionally, the fused features are directly fed into a reinforcement learning system to generate control strategies, achieving end-to-end autonomous driving. This approach enhances the accuracy and robustness of navigation systems in handling real-time decision-making requirements in complex and dynamic environments.
- *Lattice-graph-based motion planning:* An integration of a lattice-graph-based motion planning algorithm with dueling DQN is presented. This innovative combination optimizes action selection by considering the kinematic constraints of vehicles, facilitating effective navigation in unfamiliar terrains and enhancing the rationality of the action selection process with a dense reward function.
- *Dual-training paradigm:* A dual-training paradigm is introduced, integrating curriculum learning and modal separation learning to tailor the training process according to diverse task requirements. This approach capitalizes on bespoke scenarios, representing the progression from simpler to more complex learning tasks inherent in curriculum learning. By combining curriculum learning with modal separation techniques, the paradigm effectively addresses the challenges associated with multi-modal learning, as confirmed by empirical validation through comprehensive experiments.

## II. Related Works

Traditional navigation techniques, heavily reliant on pre-mapped environments and SLAM for vehicle localization, excel in static contexts but encounter challenges in dynamic, unstructured settings. These challenges stem from high computational demands and the necessity for continual map updates, leading to accumulative inaccuracies that compromise system reliability, especially in critical applications such as emergency rescue or autonomous exploration [15], [16].

These conventional systems, encompassing perception, decision-making, path planning, and control modules, struggle to adapt to swiftly changing conditions [17]. The lag in updating maps and vehicle positions hampers the system's capacity for timely navigation decisions, a situation worsened by the dependency on pre-mapped data. Traditional approaches also grapple with the integration of real-time sensory feedback, a key aspect of navigating through complex terrains, frequently resulting in delayed or less-than-optimal decisions.

Path planning, a vital element of navigation, faces obstacles due to the impracticality of global strategies in environments with unreliable GPS or incomplete sensor data, requiring constant updates that hinder real-time navigation [18]. Moreover, local path planning, although intended to utilize real-time data, encounters difficulties in intricate scenarios, constrained by the need for extensive parameter tuning and a reliance on prior knowledge, underscoring the demand for more versatile and responsive strategies.

Additionally, traditional control mechanisms, closely linked to the planning phase, often lead to less effective maneuvers in challenging environments. This reveals a fundamental issue in traditional navigation frameworks' proficiency in managing unpredictable terrains with efficiency.

Considering the extensive challenges associated with perception, planning, and control in traditional navigation frameworks, our research pivots to pioneering navigation strategies that integrate multi-modal sensory data with reinforcement learning principles. This shift aims to robustly counter the constraints of conventional methods, offering improved real-time responsiveness and adaptability. Such an approach facilitates dynamic interpretation of environmental conditions and supports informed navigation decisions, independent of pre-mapped data.

In recent explorations of navigation within unknown terrains, a distinction has been made between single-modal and multi-modal navigation methods, categorized by their approach to environmental perception. Navigation methods, which can be classified based on the patterns derived from environmental

perception data, can be broadly categorized into two groups: single-modal and multi-modal navigation.

Ma et al. [19] introduced a learning-based map-less motion planner that uses RGB images as visual inputs. They effectively decoupled the visual feature extraction module from the reinforcement learning network, thereby enhancing sampling efficiency. However, the incorporation of RGB images increases the complexity of Sim2Real transfer. Loquercio et al. [20] proposed a data-driven methodology by devising a fast eight-layer residual network. This approach generates two outputs for each input image; one guides the drone in obstacle avoidance while maintaining navigation heading, and the other enables the drone to recognize hazardous situations and respond promptly with collision probability assessments. However, the use of single-frame images raises concerns about information retention. Tai et al. [8] employed laser radar to obtain sparse 10-dimensional distance and used this distance and the target's position relative to the robot's coordinate system as inputs. Continuous steering commands were generated as outputs for end-to-end training. Fan et al. [9] realized the safe navigation of various mobile platforms in complex environments by relying exclusively on laser radar input, thus mitigating the effect of the Sim2Real gap. However, due to the sparsity of laser radar data, the method can only perceive information up to a certain height, which limits its capability to handle complex environments with obstacles of arbitrary heights and shapes.

The limitations of traditional single-modal navigation methods, which typically rely on a single type of sensor input, underscore the need for a shift towards multi-modal approaches. These approaches integrate data from multiple sensors, offering a more comprehensive understanding of the environment and improving adaptability and decision-making in navigation.

Multi-modal navigation has been extensively studied, with various methodologies explored across several studies [12], [21], [22], [23]. Sobh et al. [21] proposed a comprehensive deep-learning navigation framework that integrates image and LiDAR data. The experimental findings validated the framework's optimal performance, achieved through the fusion of probability graph models and semantic segmentation. Kalenberg et al. [22] introduced a multi-modal sensor fusion approach that combines data from grayscale and depth cameras into a unified multiheaded convolutional neural network (CNN), aiming to establish a robust navigation strategy. Xiao et al. [23] conducted an analysis to determine if the amalgamation of RGB and depth modalities can yield a more effective, end-to-end, AI-driven program compared with relying solely on a single modal. Huang et al. [12] proposed a navigation method based on multi-modal perception and deep reinforcement learning, utilizing RGB images to identify drivable areas and adopting LiDAR data for obstacle avoidance. However, this method does not consider road surface roughness, and the associated costs of LiDAR-based obstacle avoidance are high. Moreover, the methods previously mentioned rely primarily on single-frame inputs, which inherently limits their ability to capture dynamic temporal dependencies essential for understanding complex motion patterns in navigation. This results in a potential loss of crucial temporal information that is vital for predicting future states effectively.

Fig. 1. The upper section depicts the rugged terrain of the wilderness. The two images on the LHS respectively illustrate the off-road driving scenarios of an off-road vehicle and a rescue vehicle in the wilderness. On the RHS, real outdoor scenes of deep pits and water ponds are presented, while the lower section showcases boulders, pillars, and tree trunks in the outdoor environment. The central image represents the simulation environment constructed based on the complex wilderness terrain. The circular concave shape symbolizes deep pits and water ponds, protruding slopes signify boulders, and the brown rectangular prism corresponds to pillars and tree trunks. Given the depth-based nature of our simulation, no specific requirements are imposed on color or shape, and other obstacles can also be represented accordingly.

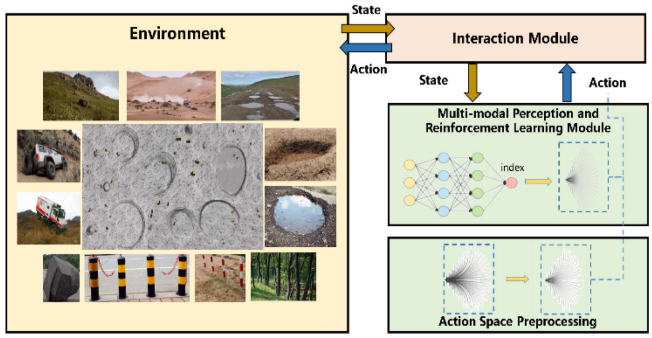



Fig. 2. System architecture.

Although existing single-modal and multi-modal navigation methods offer various solutions for the navigation of UGVs in unknown environments, they still face challenges when dealing with highly dynamic and irregular terrains. In particular, these methods exhibit limitations in the comprehensive utilization of perceptual information, maintenance of long-term memory, and adaptability in changing environments. Furthermore, existing approaches often simplify or redundantly approach action selection strategies, failing to fully consider the kinematic constraints of UGVs, thus limiting the efficiency and safety of navigation in complex terrains. These unresolved issues have inspired us to develop a new multi-modal perception and memory-guided strategy, allowing to enhance the navigational performance of UGVs in complex terrains through improved perceptual capabilities and decision-making mechanisms.

## III. Methodology

The system architecture is illustrated in Fig. 2. Similar to a typical reinforcement learning framework, our system encompasses elements such as the environment, agent, and actions. The simulation environment is meticulously designed

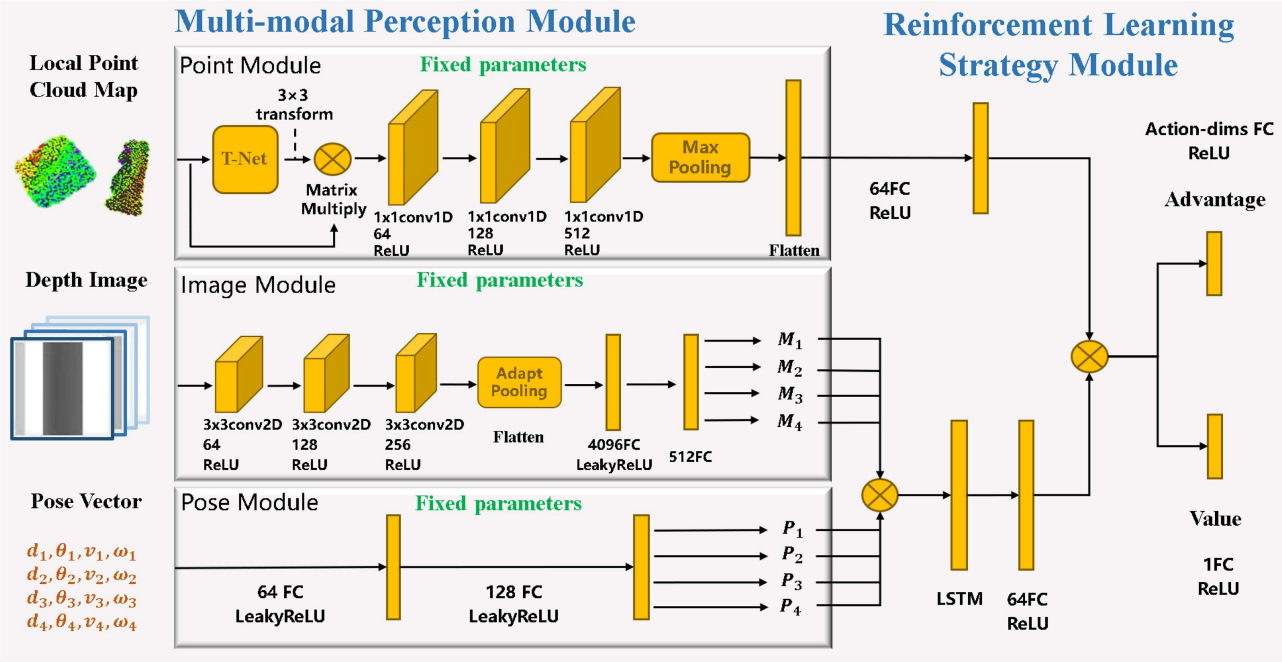



Fig. 3. The framework based on multi-modal perception and reinforcement learning.

to replicate real-world conditions. The multi-modal perception and reinforcement learning modules collectively represent the agent, with an additional interaction module serving as a critical connector between the environment and the agent. Additionally, an action space preprocessing component is included to optimize the action space for improved training efficiency.

To elaborate, the interaction module receives crucial information—including the vehicle's position, images, and point clouds—from the reinforcement learning environment module. This data undergoes preprocessing before being transmitted to the multi-modal perception module for feature extraction. The processed information is then utilized by the reinforcement learning policy module to generate actions. These actions are subsequently fed back to the interaction module, which then relays them to the appropriate control units, enabling the vehicle to navigate and accomplish the assigned tasks effectively.

This architecture ensures a seamless flow of information and robust interaction between the environment and the agent, thereby facilitating efficient navigation and task execution.

The reinforcement learning policy module is designed to ascertain navigation strategies through the implementation of a reinforcement learning method. Given the inherent limitations of each sensing technique, strategies acquired from single-modal data lack robustness, particularly in intricate scenarios such as outdoor environments. Consequently, as shown in Fig. 3, a multi-modal fusion framework grounded in the reinforcement learning paradigm for UGV navigation across challenging outdoor terrains is introduced. This comprehensive framework comprises a multi-modal perception module and a reinforcement learning policy module. The multi-modal perception module integrates diverse sensing modalities, including pose, point clouds, and depth maps, as input. The reinforcement learning policy module assimilates information from these modalities to generate linear and angular velocities as outputs. These velocities are then inputted into the UGV controller to facilitate autonomous navigation and the accomplishment of specific tasks.

### A. Reinforcement Learning

*1) Problem Formulation:* We define the reinforcement learning problem as a partially observable Markov decision process represented by the tuple $(S,\ A,\ P,\ R,\ \gamma)$, where $S$ denotes a finite set of states, $A$ represents a finite set of actions, $P$ stands for the state transition probability, $R$ is the reward function, and $\gamma \in [0, 1]$ is the discount factor. The agent, serving as a learner and decision maker, engages with the environment, which encompasses all entities interacting with the agent except the agent itself. This interaction occurs at discrete time steps within a sequence, denoted as $\mathrm{t} = 0,\ 1,\ 2,\ 3,\ \cdots$. At each time point $t$, the agent receives the environment state $s_t \in S$ and selects the action $a_t \in A$. After a time step, the agent receives a numerical reward $r_{t+1} \in R$ and transitions to the new state $s_{t+1}$. The task is formulated as a finite-horizon problem with a duration of $T$ time steps. The agent's primary objective is to acquire the optimal control policy, which is denoted as $\pi(A_t|S_t;\theta_\pi)$, ultimately maximizing the expected return.

$$\pi^* = \arg\max_{\theta_\pi} E\left[\sum_{t=0}^{T-1} \gamma^t R_{t+1}\right] \tag{1}$$

In this context, the discount factor $\gamma$ is set to 0.9, and $\mathrm{E(X)}$ represents the expectation of random variable X.

Various classical reinforcement learning algorithms, including proximal policy optimization [24], deep deterministic policy gradient [25], and DQN [26], have been developed. A noteworthy addition to this list is the dueling DQN algorithm [13], which proposes a novel neural network architecture known as

the dueling network. Although it shares similarities with the DQN algorithm in terms of using deep neural networks to approximate Q-values, the dueling DQN algorithm differs in its output structure. The algorithm's output has two branches: the scalar state value (V) for the given state and the advantage value (A), which is a vector with the same dimension as the action space, for each action. This innovative structure is formally expressed as

$$Q(s,a;\theta,\alpha,\beta) = V(s;\theta,\beta) + A(s,a;\theta,\alpha) \tag{2}$$

where $s$ denotes the state, $a$ represents the action, and the network parameters are denoted by the symbols $\alpha$, $\theta$, and $\beta$.

Equation (2) is considered unidentifiable because it does not allow for the unique recovery of V and A given Q. To address this identifiability challenge, the authors introduce forward mapping in the last module of the network. For enhanced practical effectiveness and stability, the authors also transform the maximization form into an averaging form, ultimately computing the $Q$ value, as shown in (3).

$$Q(s,a;\theta,\alpha,\beta) = V(s;\theta,\beta) + \left( A(s,a;\theta,\alpha) - \frac{1}{|A|}\sum_{a'} A(s,a';\theta,\alpha) \right) \tag{3}$$

In this study, the dueling DQN algorithm is utilized to implement a multi-modal fusion paradigm for agent training.

*2) Reinforcement Learning Setup:*

*a) Observation space:* Firstly, the pose, denoted as $P_o$, encompasses four variables: relative distance ($d$) to obstacles, relative angle ($\theta$), current velocity ($v$), and angular velocity ($\omega$) at present.

$$P_O = [d, \theta, v, \omega] \tag{4}$$

Secondly, the sensory input, denoted as $I_M$, assumes a pivotal role in obstacle avoidance tasks. Compared with point cloud information, images have higher density and a more precise representation of observed obstacles. Consequently, in this study, image data is used to accomplish obstacle avoidance tasks. The images are further categorized into RGB images and depth maps. Although RGB images provide intuitively interpretable information, challenges arise in accurately simulating real-world conditions in virtual environments because of texture, lighting, perceptual noise, and other factors. These challenges may impede the transferability of learned strategies from virtual to real-world scenarios. Hence, depth maps are selected as the input for the image module. Additionally, to mitigate the forgetting phenomenon associated with agents relying solely on single-frame images, the past three frames of images are incorporated into the LSTM module. The input image size is set to 100 × 100, which is achieved by horizontally extracting regions via area interpolation and vertically selecting the middle 100 rows of pixels.

Thirdly, the precision of Sensing Data significantly influences the learning process in reinforcement learning. Point cloud data ($P_c$) directly collected from the LiDAR sensor may lack the necessary precision. To enhance this accuracy, the 3D LiDAR SLAM algorithm, A-LOAM, is initially employed for mapping, with the outcomes stored as Point Cloud Data (PCD) files. This storage facilitates the real-time extraction of local PCD. In the preprocessing phase, a CropBox filter dynamically extracts local PCD within a rectangular prism, centered on the centroid of the vehicle, with dimensions of 5 m × 5 m × 6 m. Furthermore, the StatisticalOutlierRemoval filter is applied to remove outlier data points. To ensure a consistent input size for deep reinforcement learning methods while preserving the distinctive features of the current time step, the farthest-point sampling algorithm is employed.

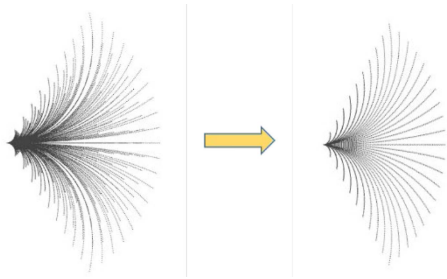


Fig. 4. Optimizing action space.

In summary, the expression for the state space is as

$$O_b = \left[P_O^9, P_O^6, P_O^3, P_O^0, I_M^9, I_M^6, I_M^3, I_M^0, P_C\right] \tag{5}$$

where $P_O^i$ denotes the pose information from the i$^{\text{th}}$ preceding frame and $I_M^i$ signifies the image information captured in the i$^{\text{th}}$ preceding frame.

*b) Action space:* In this study, the control space of the vehicle consists of two components: linear and angular velocities. In the traditional action space, actions are defined by combinations of linear and angular velocities (v, w), where the linear velocity ranges from [0, 1] and the angular velocity ranges from [-1, 1], both with a resolution of 0.1. This results in 231 different actions. Training with such a large action space increases computational complexity and training time, lowering learning efficiency. Moreover, this configuration results in a notable overlap between certain actions, which is detrimental to effective training.

To address this issue, we propose the "Lattice-graph-based motion planning" method. The primary purpose is to reduce the action space in reinforcement learning, thereby enhancing training efficiency. This method reduces the action space to a set of representative actions that effectively cover the vehicle's possible motion trajectories while maintaining diversity. This reduction not only simplifies the action space but also helps to expedite convergence. The pseudocode for generating the action space is outlined in Algorithm 1.

The Fig. 4 shows the effect of the algorithm on optimizing the action space (comparison of trajectories generated for all actions in the action space over the next 1 second).

*c) Reward sesign:* Considering that achieving autonomous navigation in rough terrains represents a multi-objective optimization challenge, five distinct rewards are devised to facilitate reinforcement learning training. These

**Algorithm 1:** Action Space Generation Algorithm Based on Lattice-Graph Sampling.

**Input:** Range and resolution of linear velocity $v_i$ and angular velocity $\omega_j$ - ($v_{min}$, $v_{max}$, $v_{res}$), ($\omega_{min}$, $\omega_{max}$, $\omega_{res}$)

**Output:** Action space $A_s$ .

Initialize the time $t \leftarrow 0$, the prediction time $T_{pre} \leftarrow 1.0$, the time interval $\triangle t \leftarrow 0.01$, the index value *ind* $\leftarrow 0$, the overlap search radius *overlap_radius* $\leftarrow 0.05$, the overlap threshold *overlap_threshold* $\leftarrow 0.95$.
Grouping by linear velocity $v_i$, the number of trajectories in each group is *group_trajectory_number* $\leftarrow (\omega_{max}$ - $\omega_{min})$ / $\omega_{res} + 1$.
**for** $v_i$ **from** $v_{min}$ **to** $v_{max}$ **step** $v_{res}$ **do**
  **for** $\omega_j$ **from** $\omega_{min}$ **to** $\omega_{max}$ **step** $\omega_{res}$ **do**
    Initialize the single-path point cloud $P_s$
    Initialize the single-point point cloud $P_t$
    **while** $t \leq T_{pre}$ **do**
      $P_t \leftarrow$ Prediction (vehicle kinematics, $(v_i, \omega_j)$)
      $t \leftarrow t + \triangle$t
      Store $p_t$ in $p_s$
      *mapping_m* $\leftarrow$ (ind, $(v_i, \omega_j)$)
    Store $p_s$ in *vector_p*
    $t \leftarrow 0$
    *ind* $\leftarrow$ *ind* $+ 1$
  // Traverse trajectories from groups other than one's own.
  **for** i **from** 0 **to** (*length(vector_p)*–group_trajectory_number) **do**
    **for** j **from** 0 **to** *length(vector_p)* **do**
      // Determine whether i and j are in different groups.
      **if** i and j are in different groups **then**
        Store *vector_p[j]* in $p_{data}$.
    **for** k **from** 0 **to** *length(vector_p[i])* **do**
      Kdtree.setInputCloud ($p_{data}$)
      (*nearest_point, nearest_distance*) $\leftarrow$ KdtreeSearch(*vector_p[i][k]* )
      **if** *nearest_distance* $<$ *overlap_radius* **then**
        *overlap_point_num* $\leftarrow$ *overlap_point_num* $+ 1$
      $\alpha =$ *overlap_point_num* / *length(vector_p[i])*
    **if** $\alpha >$ *overlap_threshold* **then**
      Store *k* in *remove_index_vector*
**for** i **from** 0 **to** (*vector_p.size* – group_trajectory_number) **do**
  **if** i **not in** *remove_index_vector* **then**
    Store i in *preserve_index_vector*
**for each** element $\in$ *preserve_index_vector* **do**
  **if** element **in** *mapping_m.index* **then**
    Store *mapping_m.value* in $A_s$
**return** the action space $A_s$

rewards include the reward for proximity to the endpoint ($r_{dis}$), the reward for reaching the endpoint ($r_{target}$), the penalty for close proximity to obstacles ($r_{obs}$), the penalty for colliding with obstacles ($r_{col}$), and the penalty for pitch and roll ($r_{rp}$). These rewards are explained in detail below.

- Reward for Proximity to the Endpoint ($r_{dis}$)

$$d_t^{index} = \text{int}\left(\frac{d_t^{target}}{w}\right) \tag{6}$$

$$d_{t-1}^{index} = \text{int}\left(\frac{d_{t-1}^{target}}{w}\right) \tag{7}$$

$$d_{sub} = d_t^{index} - d_{t-1}^{index} \tag{8}$$

$$r_{dis} = \begin{cases} \mu \times |d_{sub}|, & \text{if } (d_{sub} < 0) \\ -\mu \times |d_{sub}|, & \text{else} \end{cases} \tag{9}$$

where $\mu$ serves as a hyperparameter signifying the magnification factor, which is set to 5. $d_t^{target}$ and $d_{t-1}^{target}$ denote the current and previous time step distances between the agent and the target, respectively. $d_t^{index}$ and $d_{t-1}^{index}$ are computed by truncating the values using the "int" function in Python. **w** is another hyperparameter (set as 0.2) for ensuring that $d_{sub}$ remains within the range of [0, 1] or 2.

- Reward for Reaching the Endpoint ($r_{target}$)

$$r_{target} = \begin{cases} r_{goal}, & \text{if } (d_t^{target} < 1) \\ 0, & \text{else} \end{cases} \tag{10}$$

where $d_t^{target}$ represents the distance between the vehicle and the goal at the current time step. For training in the pose modality, $r_{goal}$ is set to 500. During training in the image and point modalities, it is adjusted to 1,000.

- Penalty for Close Proximity to Obstacles ($r_{obs}$)

$$r_{obs} = \begin{cases} -\frac{e^{5-d_{obs}}}{\mu}, & \text{if } (0.8 \leq d_{obs} < 5) \\ 0, & \text{else} \end{cases} \tag{11}$$

where $d_{obs}$ represents the minimum distance to obstacles observed in the depth map, which is set to 5. The parameter $\mu$ is introduced to prevent the penalty from being excessively large, which is set to 3.

- Penalty for Colliding with Obstacles ($r_{col}$)

$$r_{col} = \begin{cases} -200, & \text{if } (d_{obs} < 0.8) \\ 0, & \text{else} \end{cases} \tag{12}$$

- Penalty for Roll and Pitch ($r_{rp}$)

$$r_{rp} = r_{roll} + r_{pitch} \tag{13}$$

Penalty for Roll ($r_{roll}$)

$$r_{roll} = \begin{cases} 0, & \text{if } (|r| \leq 10^{\circ}) \\ -2, & \text{if } (10^{\circ} < |r| \leq 20^{\circ}) \\ -10, & \text{else} \end{cases} \tag{14}$$

Penalty for Pitch ($r_{pitch}$)

$$r_{pitch} = \begin{cases} 0, & \text{if } (|p| \leq 10^{\circ}) \\ -2, & \text{if } (10^{\circ} < |p| \leq 20^{\circ}) \\ -10, & \text{else} \end{cases} \tag{15}$$

In the expressions above, *r* and *p* respectively represent roll and pitch angles.

The total reward function can be formulated as

$$R = r_{dis} + r_{target} + r_{obs} + r_{col} + r_{rp} \tag{16}$$

TABLE I
NETWORK STRUCTURE OF THE MULTI-MODAL PERCEPTION MODULE

| Module | Type (Activation Function) | Output Dimension | Parameter |
|---|---|---|---|
| Pose | Fully Connected Layer (LeakyReLU) | 64 | / |
| | Fully Connected Layer (LeakyReLU) | 128 | / |
| Image | Two-dimensional Convolutional Layer | 64 | Kernel size is 3. Stride is 2. Padding is 1. |
| | Two-dimensional Convolutional Layer | 128 | Kernel size is 3. Padding is 1. |
| | Two-dimensional Convolutional Layer | 256 | Kernel size is 3. Padding is 1. |
| | Adaptive Average Pooling Layer | / | 7×7 |
| | Flatten Layer | 256×7×7 | / |
| | Fully Connected Layer (LeakyReLU) | 4096 | / |
| | Linear Layer | 512 | / |
| Point | Affine Transformation Matrix Generation Layer | 3×3 matrix | / |
| | One-dimensional Convolutional Layer (ReLU) | 64 | Kernel size is 1. |
| | One-dimensional Convolutional Layer (ReLU) | 128 | Kernel size is 1. |
| | One-dimensional Convolutional Layer (ReLU) | 512 | Kernel size is 1. |
| | Max Pooling Layer | / | / |

*d) Termination conditions:*

- The number of steps exceeds 400.
- Any pixel value in the depth map that is less than 0.8 indicates a collision.
- A roll or pitch angle that is greater than 30°.

### B. Multi-Modal Perception Module

In complex outdoor environments, single-modal data processing often fails to handle variable terrain conditions effectively. Therefore, we designed a multi-modal perception module that integrates pose, depth images, and point cloud data to enhance environmental awareness and decision-making capabilities. Fig. 3 illustrates our proposed network architecture, which incorporates these three distinct modalities. Details of each modality's network parameters are shown in Table I.

*Pose Module:* The pose module acquires real-time vehicle position, velocity, and orientation information. This module is crucial in complex terrains, providing precise pose information essential for navigating complex and rugged environments. Leaky ReLU activation functions are used to avoid the gradient vanishing problem, thereby enhancing feature extraction performance.

*Depth Image Module:* The depth image module primarily facilitates obstacle avoidance. By utilizing depth information from cameras, this module enables the system to detect obstacles in front of the vehicle and adjust its path to prevent collisions in complex outdoor environments. Features are processed through three convolutional layers and an adaptive pooling layer to ensure effective obstacle avoidance across varying distances, thus enhancing the vehicle's understanding of its surroundings [27].

*Point Cloud Module:* The point cloud module primarily handles uneven surfaces and rugged terrain, especially in situations where image data may be limited or unable to capture ground details. For instance, in complex outdoor terrains such as deep pits, water ponds, and boulders, images might fail to capture distant or subtle ground features, whereas point cloud data can provide valuable supplementary information. By using an affine transformation matrix T-Net ($3 \times 3$) to normalize the point cloud data, we can reduce the impact of spatial transformations or vehicle vibrations on the data. Coupled with a max pooling layer to handle the unordered nature of point cloud data, this approach improves the system's ability to process complex terrain features like uneven surfaces and tree trunks [28].

The integration of pose, depth image, and point cloud data in the multi-modal perception module provides a comprehensive understanding of complex terrain. The pose module acquires relative information to the target point, while depth images are used for avoiding obstacles in front and point clouds handle uneven surfaces. The fusion of these three modalities enhances the system's performance in complex environments.

### C. Reinforcement Learning Policy Module

The reinforcement learning policy module takes input from the pose data, depth images, and LiDAR data provided by the perception module. Initially, it combines the output data of each frame's pose ($P_i$) with the output data of depth images ($M_i$) obtained from the multi-modal perception module through (17). In deep learning, $P_i$ serves as the label for $M_i$. Subsequently, the fused data are organized in chronological order from the earliest to the latest, as illustrated in (18). The ordered data are then inputted into the LSTM algorithm, and the specific representation process is as:

$$\Omega_1 = (M_1, P_1)$$
$$\Omega_2 = (M_2, P_2)$$
$$\Omega_3 = (M_3, P_3)$$
$$\Omega_4 = (M_4, P_4) \tag{17}$$
$$\Omega = (\Omega_1, \Omega_2, \Omega_3, \Omega_4) \tag{18}$$

The symbol $\Omega$ represents the data format inputted to LSTM.

Afterward, the features and measurements obtained from the local point cloud are integrated and processed through their respective fully connected layers, resulting in the outputs, i.e., Advantage and Value. Notably, Advantage shares dimensions

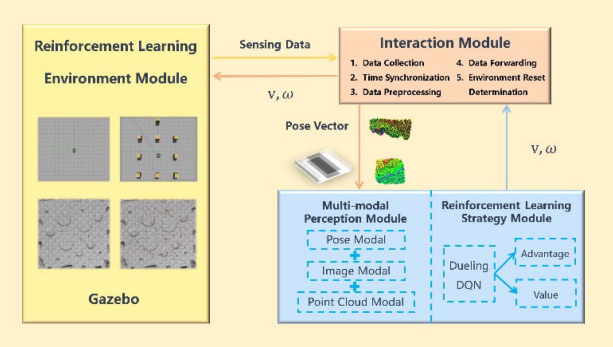



Fig. 5. Experimental setup.

with the action space and can be incorporated into the computation of the Q value as in (3).

## IV. Experimental Results

This section presents a structured exploration of our experimental results, showcasing the effectiveness and innovation of our navigation strategies through rigorous testing and comparison. Initially, the experimental setup and implementation specifics, including the system framework, training protocol, and detailed implementation nuances, are outlined, laying the groundwork for a comprehensive evaluation. The baseline methods and evaluation metrics against which our approach is compared are then introduced, setting the stage for a detailed analysis.

A focal point of our experiments is the evaluation of the Modal Separation Learning Training Paradigm, demonstrating its proficiency in learning and generalizing complex navigational tasks. Through comparative experiments conducted in varied scenarios—namely, Flat Obstacle Scenes and Rough Terrain Obstacle Scenes—we critically assess our method's performance relative to established benchmarks.

### A. Implementation Details

*1) Experimental Setup:* The system consists of four modules: the Reinforcement Learning Environment module, Interaction module, Multi-modal Perception module, and Reinforcement Learning Policy module. The interactions among these modules are depicted in Fig. 5.

Gazebo plays a crucial role in this framework by providing a simulated environment where the UGV can operate. It enables the testing and validation of navigation strategies in a controlled, repeatable setting, replicating real-world conditions such as varied terrains and dynamic obstacles.

As previously explained, the Multi-modal Perception module integrates data from various sensors to create a comprehensive understanding of the environment, while the Reinforcement Learning Policy module utilizes this information to generate effective navigation strategies.

The interaction module receives information, including the vehicle's position, images, and point clouds, from the reinforcement learning environment module. After preprocessing, it forwards the data to the multi-modal perception module for feature extraction. Subsequently, the reinforcement learning policy module outputs linear and angular velocities, which are then fed back to the interaction module. Finally, the information is transmitted to the UGV controller in Gazebo, facilitating navigation to achieve the designated tasks.

To better illustrate the significance of the Interaction module, we detail its specific processing procedure below:

1) Data Collection: The module gathers sensor data, including the vehicle's position, images, and point clouds, from the reinforcement learning environment.
2) Time Synchronization: It synchronizes the collected data to ensure consistency across different sensor modalities, which is crucial for accurate perception and decision-making.
3) Data Preprocessing: The collected and synchronized data undergoes preprocessing to filter noise and enhance relevant features, preparing it for further analysis.
4) Data Forwarding: The preprocessed data is then forwarded to the multi-modal perception module, where features are extracted and fused for comprehensive situational awareness.
5) Environment Reset Determination: The interaction module monitors the environment and determines if a reset is required, ensuring that the system can handle unexpected events and continue operating effectively. The reset conditions are detailed in Section III-A-2-d).

These steps ensure that the Interaction module processes and handles data efficiently, supporting the system's overall functionality and performance.

*2) Training Protocol:* The strategy network based on the multi-modal fusion approach faces challenges in scalability, making it difficult to learn effectively. To overcome this limitation, modal separation and a curriculum learning approach are introduced as auxiliary training techniques. The detailed training protocol is outlined in the Table II. The overall task is decomposed into four subtasks: achieving a quick endpoint arrival, reaching the endpoint while avoiding collisions, reaching the endpoint while preventing rollovers, and reaching the endpoint while avoiding both collisions and rollovers. Concurrently, four scenarios are intentionally designed to train our navigation strategy for these subtasks: flat open space, flat obstacle environment, rugged open terrain, and rugged obstacle environment. The training difficulty progressively increases across these scenarios. A modal separation paradigm is adopted to sequentially train the Pose, Pose+Image, Pose+Point, and Pose+Image+Point networks, aiming to expedite convergence.

*3) Implementation Details:* This paper involves other parameters as shown in Table III. Additionally, we conducted experiments with the LSTM model using different parameters, specifically varying the number of layers (1, 2, and 3). The results indicated that setting the number of layers to 1 achieved the best performance, which was subsequently used in our analysis.

### B. Baseline Methods and Evaluation Metrics

*1) Baseline Methods:*

- Flat Obstacle Scenes

TABLE II
SPECIFIC TRAINING PROTOCOL

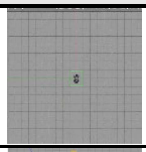
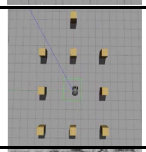
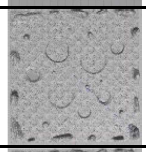
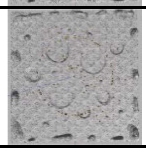
| Training Modality | Training Approach | Training Scenario | Subtask | Training reward |
|---|---|---|---|---|
| Pose | Directly train the Pose modality and strategy module as shown in Fig. 3. |  | Arrive at the destination quickly | $R_1 = r_{dis} + r_{target}$ |
| Pose+Image | Load and fix network parameters for the Pose modality, then train the Image modality and strategy module. |  | Arrive at the destination quickly while avoiding collisions | $R_2 = r_{dis} + r_{target} + r_{obs} + r_{col}$ |
| Pose+Point | Load and fix network parameters for the Pose modality, then train the Point modality and strategy module. |  | Arrive at the destination quickly while avoiding overturning | $R_3 = r_{dis} + r_{target} + r_{rp}$ |
| Pose+Image+Point | Load and fix network parameters for the Pose, Image, and Point modalities, then proceed to train only the strategy module. |  | Arrive at the destination quickly while avoiding both collisions and overturning | $R = r_{dis} + r_{target} + r_{obs} + r_{col} + r_{rp}$ |

TABLE III
TRAINING PARAMETERS

| | | |
|---|---|---|
| **LSTM Algorithm Parameter Configuration** | Number of Layers | 1 |
| | Input Layer and Hidden Layer Dimensions | 640 |
| | Hidden States | The optimizer model is initialized as a tensor of zeros at each training iteration. |
| **Optimizer model** | Adam | |
| **Learning rate** | 0.001 | |
| **Greedy strategy coefficient** | $\varepsilon = \max(\varepsilon_{init}, \varepsilon - (\varepsilon_{init} - \varepsilon_{min}) * \varepsilon_{decay})$ The initial value of the greedy policy coefficient $\varepsilon_{init}$ is set to 0.6; the minimum value of the greedy policy coefficient $\varepsilon_{min}$ is set to 0.1; and the decay rate $\varepsilon_{decay}$ is set to 0.0001. | |
| **Batch size** | Training with Pose modal set to 1024<br>Training with Image and Point modalities set to 32 | |
| **Local map observation size** | 5m × 5m × 6m<br>(length × width × height) | |

VGG16 [29]: This is a classic convolutional neural network model consisting of 13 convolutional layers and 3 fully connected layers, with a depth of 16.
ResNet18: The basic architecture of this network is based on ResNet [30], and its depth is 18 layers.
ResNet50: The basic architecture of this network is based on ResNet, and its depth is 50 layers.
DifeNet (without LSTM): This is a streamlined version of our method that uses only single-frame images and poses as inputs, removing the LSTM module.

- Rough Terrain Obstacle Scenes
  Laser-only [8]: This is a reinforcement learning-based method that relies solely on lidar data, ensuring that the moving robot can avoid collisions with obstacles.
  RGB-only [19]: This is a deep learning-based method that uses RGB images to predict control strategies.
  Pose+Image: This is a simplified version of our method, using only depth images as network inputs, without considering local point cloud data.
  Pose+Point: This is another simplified version of our method, using only local point cloud data as network inputs, without considering depth images.

*2) Evaluation Metrics:*

- Success Rate: The probability of successfully reaching the endpoint in all test episodes.
- Overturn Rate: The likelihood of rollovers or pitch angles exceeding 30° in all test episodes.
- Collision Rate: The chance of collisions with obstacles or slopes in all test episodes.
- Average Speed: Indicates whether the model considers speed while avoiding collisions or rollovers.
- Average Distance in Successful Episodes: Reflects whether the model considers distance factors when successfully reaching the endpoint.
- Single Step Reward: Measures the effectiveness of our reward settings.
- Average Episode Reward: Represents the model's overall performance across all test episodes.

TABLE IV
TRAINING COMPARISON

| Training Method/Metric | Convergence Time | Convergence Value |
|---|---|---|
| Non-separated modality | 4.069h | 555.2 |
| Modality separation (ours) | **3.409h** | **947.2** |

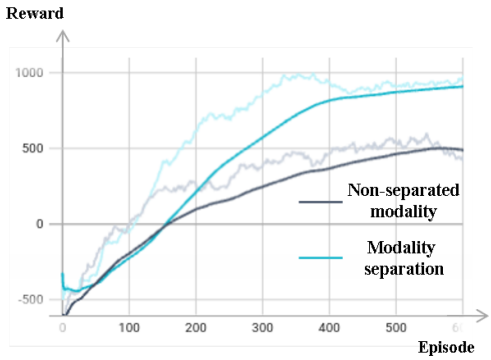



Fig. 6. Reward curves for the two training approaches.

### C. Evaluating the Modal Separation Learning Training Paradigm

Modality separation training offers several advantages. By isolating and training each sensory modality independently, we can ensure that each module reaches optimal performance without interference from others. This approach not only improves the efficiency of the training process but also enhances the overall stability and effectiveness of the navigation system.

The training scene images are presented in Table II. Initially, in the flat and open terrain scene, the Pose module is trained until convergence. Subsequently, in the flat obstacle scene, the parameters of the Pose module are loaded and fixed, and training for the Image module commences. In the rough and open terrain scene, the parameters of the Pose module are loaded and fixed, and training for the Point module is initiated. Lastly, in the rough obstacle scene, the parameters of all three modalities are loaded and fixed, and only the reinforcement learning policy module is trained.

To demonstrate the effectiveness of this method, we compared it with the non-separated modality training (where no parameters are fixed, and all three modalities along with the reinforcement learning policy module are trained simultaneously) in the rough obstacle scene. The results are as follows:

Table IV demonstrates that the modality separation training method achieves superior convergence time and values compared to the non-separated modality training method. This is further corroborated by Fig. 6, which presents the reward curves for both training approaches. The detailed comparison in Table IV highlights the faster convergence and enhanced performance of the modality separation method, affirming its superior learning efficiency and robustness.

Furthermore, for a more intuitive examination of the impact of each modal, ablation experiments on three modalities are conducted. The results of these experiments are detailed in Table V.

Based on the data presented in Table V, in flat and open scenes, the Pose network achieved a navigation success rate of 100% during testing, affirming the stability of the Pose modal and laying a robust foundation for the subsequent integration of additional modalities. Furthermore, in flat terrain obstacle scenes, the Pose+Image network demonstrated an 81% increase in navigation success rate compared to the Pose network. In rugged and open scenes, the Pose+Point network exhibited an 80% improvement in navigation success rate relative to the Pose network. These findings individually underscore the efficacy of the Image and Point modalities.

### D. Comparative Experiments

- Flat Obstacle Scenes
  In the context of processing depth images, our designed network, DifeNet, has been compared with established networks such as VGG16, ResNet18, and ResNet50. To ensure consistency in comparison, the first layer of these three classical networks is standardized to match the first layer of our network. The comparative results are presented below: The outcomes derived from Fig. 7 and Table VI elucidate that, despite DifeNet (without LSTM) exhibiting a slower convergence rate than the VGG16 algorithm, its convergence peak surpasses that of VGG16 by more than twofold. Furthermore, throughout the training phases spanning 0 to 600 episodes, as well as in terms of final convergence episodes and values, DifeNet showcases superior performance compared to ResNet18 and ResNet50. Additionally, the comparative analysis between DifeNet and its ablated version, DifeNet (without LSTM), specifically in terms of convergence values, serves to validate the efficacy of our LSTM module. Consequently, it can be inferred that, in contrast to the traditional algorithms such as VGG16, ResNet18, and ResNet50, the DifeNet network we designed exhibits commendable performance in processing depth images.
- Rough Terrain Obstacle Scenes
  To demonstrate the effectiveness of our proposed multimodal reinforcement learning approach, we designed several typical rough terrain scenarios and compared several well-established reinforcement learning approaches. As shown in Fig. 8, in Scenario (a), solely impassable boulders (yellow-green circles) are present, with the primary objective of evaluating the smoothness of paths generated by each model. Scenario (b) introduces both impassable boulders and randomly generated obstacles (blue-green cubes), serving to assess the obstacle avoidance capabilities of each model. Scenario (c) features a singular pit (red circle), strategically designed to examine the models' proficiency in navigating around such depressions. In Scenario (d), a combination of pits and randomly generated obstacles is presented, aiming to comprehensively evaluate the performance of each model. Lastly, Scenario (e) closely resembles Scenario (d) with the distinction that pits in Scenario

TABLE V
EXPERIMENTAL RESULTS OF MODAL SEPARATION LEARNING

| Evaluated Modality | Scenario | Experimental Setup | Experimental Results | |
|---|---|---|---|---|
| | | | Evaluated Network | Navigation Success Rate |
| Pose | Flat and Open Terrain Scenario | Performing 100 Tests with Randomly Selected Initial Positions for Each Iteration | Pose | 100% |
| Image | Flat Terrain Obstacle Scenario | Performing 100 Tests with a Fixed Random Seed to Ensure Consistency in the Testing Environment | Pose | 14% |
| | | | Pose+Image | 95% |
| Point | Rugged and Open Terrain Scenario | Fixing the Random Seed and Testing with 10 Initial Points Aligned along the Centerline on a Rugged Terrain Surface | Pose | 10% |
| | | | Pose+Point | 90% |

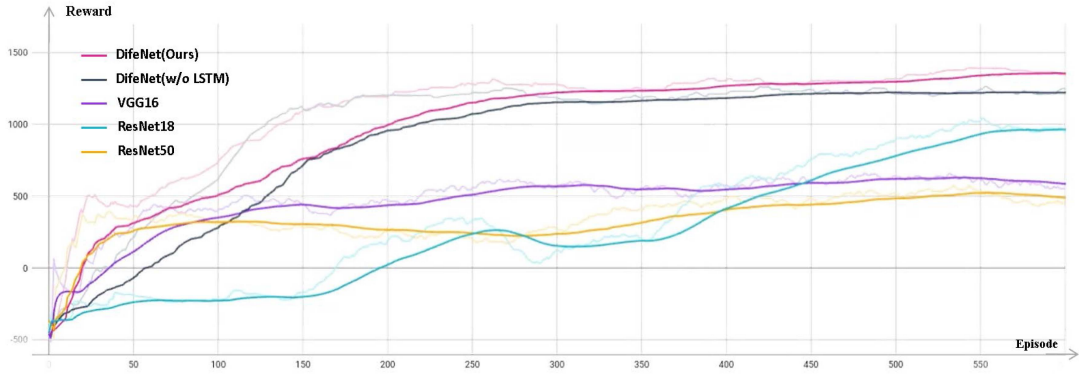



Fig. 7. Training comparative experiments.

TABLE VI
TRAINING PROGRESS

| Algorithm | Training Progress Statistics for Episodes 0 to 600 | | | | | Convergence Episodes | Convergence Metrics |
|---|---|---|---|---|---|---|---|
| | Min | Max | Start Value | End Value | △Value | | |
| VGG16 | -519.4 | 660.4 | -459.2 | 553.2 | ↑1012.4 | **290** | 573.8 |
| ResNet18 | -454.1 | 1050 | -454.1 | 946.5 | ↑1400.6 | 578 | 963.5 |
| ResNet50 | -499.6 | 577.4 | **-366.7** | 455 | ↑821.7 | 552 | 531.6 |
| DifeNet (without LSTM) | **-406.6** | 1265 | -406.6 | 1250 | ↑1656.6 | 317 | 1159 |
| DifeNet (Ours) | -473.5 | **1394** | -473.5 | **1350** | **↑1823.5** | 594 | **1364** |

d were part of the training data, while those in Scenario (e) were not, thereby serving as a test for the generalization capability of our algorithm. According to the findings in Table VII, when success rates, rollover rates, and collision rates are held constant, the Pose+Image model outperforms Laser-only in terms of both single-step rewards and average speed. This observation underscores the efficacy of our action space design, a validation further supported by Fig. 8(a). In relation to collision rates, the Pose+Image approach exhibits superiority over RGB-only, affirming the effectiveness of our image-based obstacle avoidance method, as evidenced in Fig. 8(b) where the model adeptly navigates through the space between two obstacles while maintaining a secure distance.

Concerning rollover rates, our Pose+Point model surpasses Laser-only, RGB-only, and Pose+Image. This highlights the efficacy of our point cloud module, despite occasional situations illustrated in Fig. 8(c). Subsequent joint training with Pose+Image+Point yields favorable results in this scenario. Fig. 8(d) visually demonstrates the effectiveness of Pose+Image+Point in seamlessly navigating through a pit and obstacles to reach the destination.

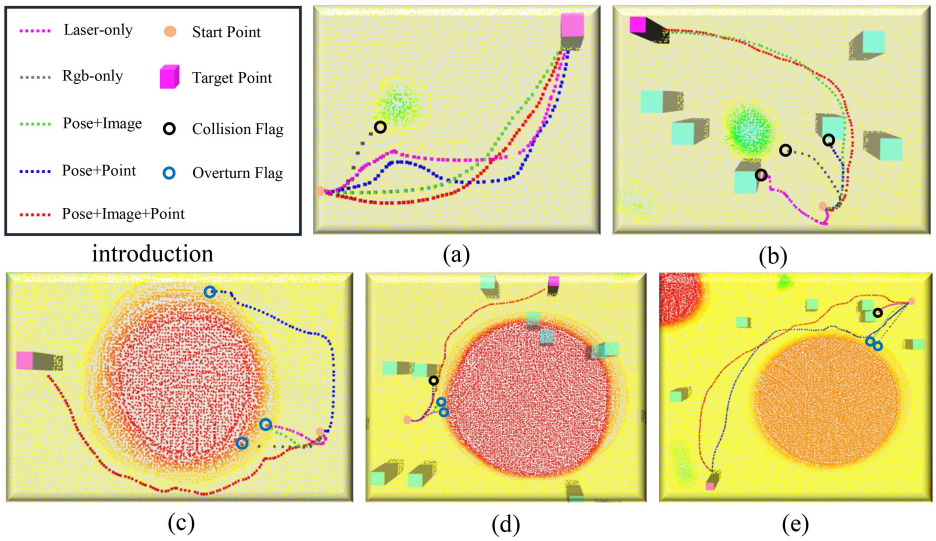



Fig. 8. Visual representation of experimental results.

TABLE VII
STATISTICAL ANALYSIS OF EXPERIMENTAL OUTCOMES

| Algorithm | Success Rate | Overturn Rate | Collision Rate | Average Speed (m/s) | Average Distance in Successful Episodes (m) | Single Step Reward | Average Episode Reward |
|---|---|---|---|---|---|---|---|
| Laser-only | 0.3 | 0.6 | 0.1 | 0.858 | 16.910 | -1.337 | -1.360 |
| RGB-only | 0.2 | 0.5 | 0.3 | 0.8843 | 17.476 | -2.514 | 130.976 |
| Pose+Image | 0.3 | 0.6 | 0.1 | 0.936 | **15.631** | -0.030 | 230.478 |
| Pose+Point | 0.6 | 0.1 | 0.3 | 0.897 | 26.269 | 0.437 | 405.95 |
| Pose+Image +Point | **0.9** | **0** | **0.1** | **0.938** | 22.140 | **8.140** | **1166.058** |

Furthermore, with regard to the average length of successful episodes, although our method may not be optimal (potentially due to covering more distance when avoiding larger pits or obstacles, as indicated in Fig. 8(e)), Pose+Image+Point attains the highest success rate in navigating to the destination. Fig. 8(e) also suggests a measure of generalization capability in our approach.

## V. CONCLUSION

In this paper, an innovative Reinforcement Learning (RL) framework grounded in multi-modal perception was introduced, facilitating the smooth and secure navigation of UGV in intricate and rugged environments. Our approach incorporated curriculum learning and modal separation principles, with the design of four progressively challenging scenarios, each corresponding to a specific targeted module. This methodology circumvented the challenges associated with directly training on highly difficult scenarios, which may lead to convergence issues and complicate parameter troubleshooting. A Convolutional Neural Network - Long Short-Term Memory (CNN-LSTM) network named DifeNet was proposed for processing depth images. The novel fusion involves inputting features extracted from multi-frame poses and images into the LSTM, effectively mitigating the forgetting phenomenon associated with single-frame inputs. Rigorous experimentation in a simulated environment serves to validate the efficacy of the proposed approach. Additionally, the demonstrated effectiveness of the point cloud processing framework in extracting information from unknown rugged terrains is noteworthy.

While our method predominantly addresses local path planning challenges beyond the scope of GPS-based global navigation, future endeavors will explore the integration of global planning algorithms or expert knowledge from human operators into the learning process. Furthermore, the transition from simulation to real-world applications for physical UGVs and the realization of safe navigation for multiple vehicles in such environments [31], [32] represent crucial directions for future research.

## REFERENCES

[1] J. Wu et al., "An adaptive conversion speed Q-learning algorithm for search and rescue UAV path planning in unknown environments," *IEEE Trans. Veh. Technol.*, vol. 72, no. 12, pp. 15391–15404, Dec. 2023, doi: 10.1109/TVT.2023.3297837.

[2] W. Wang, L. Wang, J. Wu, X. Tao, and H. Wu, "Oracle-guided deep reinforcement learning for large-scale multi-UAVs flocking and navigation," *IEEE Trans. Veh. Technol.*, vol. 71, no. 10, pp. 10280–10292, Oct. 2022, doi: 10.1109/TVT.2022.3184043.

[3] K. Zhu and T. Zhang, "Deep reinforcement learning based mobile robot navigation: A review," *Tsinghua Sci. Technol.*, vol. 26, no. 5, pp. 674–691, Oct. 2021, doi: 10.26599/TST.2021.9010012.
[4] S. Josef and A. Degani, "Deep reinforcement learning for safe local planning of a ground vehicle in unknown rough terrain," *IEEE Robot. Automat. Lett.*, vol. 5, no. 4, pp. 6748–6755, Oct. 2020, doi: 10.1109/LRA.2020.3011912.
[5] B. Liang et al., "Real-time stereo image depth estimation network with group-wise L1 distance for edge devices towards autonomous driving," *IEEE Trans. Veh. Technol.*, vol. 72, no. 11, pp. 13917–13928, Nov. 2023, doi: 10.1109/TVT.2023.3284011.
[6] O. Natan and J. Miura, "End-to-end autonomous driving with semantic depth cloud mapping and multi-agent," *IEEE Trans. Intell. Veh.*, vol. 8, no. 1, pp. 557–571, Jan. 2023, doi: 10.1109/TIV.2022.3185303.
[7] P. Cai, Y. Sun, H. Wang, and M. Liu, "VTGNet: A vision-based trajectory generation network for autonomous vehicles in urban environments," *IEEE Trans. Intell. Veh.*, vol. 6, no. 3, pp. 419–429, Sep. 2021, doi: 10.1109/TIV.2020.3033878.
[8] L. Tai, G. Paolo, and M. Liu, "Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation," in *Proc.2017 IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2017, pp. 31–36.
[9] T. Fan, X. Cheng, J. Pan, D. Monacha, and R. Y ang, "Crowd-move: Autonomous mapless navigation in crowded scenarios," 2018, *arXiv:1807.07870*.
[10] K. Kang, S. Belkhale, G. Kahn, P. Abbeel, and S. Levine, "Generalization through simulation: Integrating simulated and real data into deep reinforcement learning for vision-based autonomous flight," in *Proc. 2019 IEEE Int. Conf. Robot. Automat.*, 2019, pp. 6008–6014.
[11] Y. Wu, Z. Rao, W. Zhang, S. Lu, W. Lu, and Z. Zha, "Exploring the task cooperation in multi-goal visual navigation," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, 2019, pp. 609–615.
[12] X. Huang, H. Deng, W. Zhang, R. Song, and Y. Li, "Towards multi-modal perception-based navigation: A deep reinforcement learning method," *IEEE Robot. Automat. Lett.*, vol. 6, no. 3, pp. 4986–4993, Jul. 2021, doi: 10.1109/LRA.2021.3064461.
[13] Z. Wang, T. Schaul, M. Hessel, H. van Hasselt, M. Lanctot, and N. de Freitas, "Dueling network architectures for deep reinforcement learning," in *Proc. 33rd Int. Conf. Mach. Learn.*, 2016, vol. 48, pp. 1995–2003.
[14] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.
[15] H. Xie et al., "Semi-direct multimap SLAM system for real-time sparse 3-D map reconstruction," *IEEE Trans. Instrum. Meas.*, vol. 72, 2023, Art. no. 4502013, doi: 10.1109/TIM.2023.3240206.
[16] Z. Xu, C. Wang, T. Jiang, Z. Liu, and N. Zheng, "Impediments to environmental awareness in autonomous driving systems and its effect on user adoption," *IEEE Trans. Intell. Veh.*, early access: Mar. 08, 2024, doi: 10.1109/TIV.2024.3373773.
[17] J. Zhao et al., "Autonomous driving system: A comprehensive survey," *Expert Syst. Appl.*, vol. 242, Dec. 2023, Art. no. 122836, doi: 10.1016/j.eswa.2023.122836.
[18] G. Bai, Y. Chen, X. Hu, and J. Cui, "Correction redistribution mechanism based on forward-reverse solutions and real-time path dynamic adaptive re-planning for multi-AUVs collaborative search," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 7, pp. 7583–7601, Jul. 2024, doi: 10.1109/TITS.2024.3352902.
[19] L. Ma and J. Chen, "Using RGB image as visual input for mapless robot navigation," 2019, *arXiv:1903.09927*.
[20] A. Loquercio, A. I. Maqueda, C. R. del-Blanco, and D. Scaramuzza, "DroNet: Learning to fly by driving," *IEEE Robot. Automat. Lett.*, vol. 3, no. 2, pp. 1088–1095, Apr. 2018, doi: 10.1109/LRA.2018.2795643.
[21] I. Sobh et al., "End-to-end multi-modal sensors fusion system for urban automated driving," in *Proc. Neural Inf. Process. Syst.*, 2018, pp. 1–9.
[22] K. Kalenberg et al., "Stargate: Multimodal sensor fusion for autonomous navigation on miniaturized UAVs," *IEEE Internet Things J.*, vol. 11, no. 12, pp. 21372–21390, Jun. 2024, doi: 10.1109/JIOT.2024.3363036.
[23] Y. Xiao, F. Codevilla, A. Gurram, O. Urfalioglu, and A. M. López, "Multimodal end-to-end autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 1, pp. 537–547, Jan. 2022, doi: 10.1109/TITS.2020.3013234.
[24] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.
[25] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.
[26] V. Mnih et al., "Playing atari with deep reinforcement learning," 2013, *arXiv:1312.5602*.
[27] Y. Wang, H. He, and C. Sun, "Learning to navigate through complex dynamic environment with modular deep reinforcement learning," *IEEE Trans. Games*, vol. 10, no. 4, pp. 400–412, Dec. 2018, doi: 10.1109/TG.2018.2849942.
[28] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 652–660.
[29] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.
[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.
[31] Y. Xue and W. Chen, "Multi-agent deep reinforcement learning for UAVs navigation in unknown complex environment," *IEEE Trans. Intell. Veh.*, vol. 9, no. 1, pp. 2290–2303, Jan. 2024, doi: 10.1109/TIV.2023.3298292.
[32] J. Wang, X. Wang, X. Liu, C. -T. Cheng, F. Xiao, and D. Liang, "Trajectory planning of UAV-enabled data uploading for large-scale dynamic networks: A trend prediction based learning approach," *IEEE Trans. Veh. Technol.*, vol. 72, no. 6, pp. 8272–8277, Jun. 2023, doi: 10.1109/TVT.2023.3242272.

**Zhixuan Han** is currently working toward the Ph.D. degree with the School of Transportation Science and Engineering, Beihang University, Beijing, China. His research interests include decision making, trajectory planning, and trajectory tracking control of intelligent vehicles.

**Peng Chen** (Member, IEEE) received the Ph.D. degree from Nagoya University, Nagoya, Japan, in 2012. He is currently an Associate Professor with the School of Transportation Science and Engineering, Beihang University, Beijing, China. His research interests include intelligent transportation systems, urban traffic operation and control, decision making, trajectory planning, and trajectory tracking control of intelligent vehicles. He is a member of IEEE ITS Society.

**Bin Zhou** received the Ph.D. degree with the school of transportation science and engineering from Beihang University, Beijing, China, in 2018. He is currently an Assistant Professor with the School of Transportation Science and Engineering, Beihang University, Beijing. His research interests include deep-learning and intelligent vehicle.

**Guizhen Yu** received the Ph.D. degree from Jilin University, Changchun, China, in 2003. He is currently a Full Professor and Directory with the School of Transportation Science and Engineering, Beihang University, Beijing, China. His research interests include intelligent vehicle and urban traffic operations.