Multi-Modal Attention Perception for Intelligent Vehicle Navigation using Deep Reinforcement Learning

Zhenyu Li, *IEEE Member*, Tianyi Shang, Pengjie Xu, Wenhao Pei

***Abstract*—In this paper, we propose a new framework for collision-free intelligent vehicle navigation, aiming to successfully avoid obstacles using Deep Reinforcement Learning (DRL). The navigation system separates perception and control and utilizes multi-modal perception to achieve reliable online interaction with the surroundings. This allows for direct policy learning to generate flexible actions and avoid collisions. Our navigation system establishes a connection between the virtual environment and the real world, allowing learning policies in the virtual environment to be implemented in real-world environments through transfer learning. Our approach aims to integrate camera, Lidar, and IMU data to construct a multi-modal perception-based environment model, which is a state input for reinforcement learning. In this process, we utilize a series of Cross-Domain Self-Attention (CDSAttention) layers to enhance visual and LiDAR perception, promoting significant improvements in perception. We also introduce the Recurrent Deduction (RD) to facilitate global decision-making based on local perception. Additionally, we introduce the Self-Assessment Gradient model (SAGM) into the DRL process to further optimize the learning policy. The experimental results demonstrate the proposed approach reduces the disparity between the virtual environment and the real world, highlighting its superiority over other state-of-the-art methods. Our project page is publicly available at https://github.com/CV4RA/MMAP-DRL-Nav.**

***Index Terms*—Autonomous navigation, Multi-modal visual perception, Cross-domain attention mechanism, Deep reinforcement learning.**

I. Introduction

OVER the past years we have seen the unprecedented growth of automatic driving applications in the fields of intelligent transportation, ride-sharing, and unmanned transportation. A long-term goal of automated driving applications is to build an intelligent system that can perform various tasks without human intervention. However, the smooth realization of this goal is full of challenges, among which the changes in light and perspective will be the key factors affecting the successful completion of navigation tasks. The existing autonomous navigation schemes can be adopted including the vision-based method [1], the Lidar-based method [2], and the multi-sensor fusion method [3]. However, general multi-sensor fusion approaches are designed to simply and brutally integrate the sensing information of each sensor to complete the acquisition of redundant information that forms the lower discriminative features. We propose to use a series of self-attention models to make the navigation system pay more attention to the important clues.

Existing research, such as [4], focuses on path planning using local maps but overlooks the interaction between the vehicle and the environment. In recent years, deep reinforcement learning (DRL) has gained rapid traction in the field of autonomous driving due to its high level of environmental interactivity, which has led to significant improvements in interaction relations. DRL stands for Deep Reinforcement Learning, which is a type of machine learning (ML). Compared with traditional supervised learning [5] and unsupervised learning [6], DRL has the characteristic of learning from interactions and does not require a large amount of labeled data. Previous studies have primarily concentrated on Deep Q-Network (DQN) for Deep Reinforcement Learning (DRL), which allowed the agent to combine deep learning (DL) with decision-making DRL for the first time [7], [8]. To further facilitate interaction, the deep deterministic policy gradient (DDPG) is presented as a solution to handle continuous and high-dimensional actions [9], [10].

However, we all know that training agents using reinforcement learning involves a continuous cycle of exploration, trial and error, and further exploration. This process can be very dangerous to implement in a real-world environment where real people and vehicles are present. Therefore, the optimal approach is to train agents in virtual environments by creating various types of scenarios, and then transferring the learned policy from the virtual environments to the real environment for vehicle navigation [11], [12]. In our work, we propose a novel recurrent deduction deep reinforcement learning model to mitigate the disparity between the virtual and real environments during the policy transition process. The contributions of the proposed approach are summarized as follows: 1) We propose a multi-modal perception-based approach to promote the performance of vehicle navigation by a recurrent deduction model. 2) We introduce the Cross-

This work was funded by the Natural Science Foundation of Shandong Province (R2024QF284), the Qing Chuang Plan by the Department of Education of Shandong Province (24240904; 24240902), the Chinese Society of Construction Machinery Young Talent Lifting (CCMSYESS2023001), the Opening Foundation of Key Laboratory of Intelligent Robot (HBIR202301), the Fujian Key Laboratory of Spatial Information Perception and Intelligent Processing (FKLSIPIP1027). (*Corresponding authors: Zhenyu Li*).

Zhenyu Li and Wenhao Pei are with the School of Mechanical Engineering, Qilu University of Technology (Shandong Academy of Sciences), Jinan 250353, China, (e-mail: lizhenyu@qlu.edu.cn).

Tianyi Shang is with the School of Electronic Information Engineering, Fuzhou Technology, Fuzhou 350108, China (e-mail: 832201319@fzu.edu.cn).

Pengjie Xu is with the School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai 200030, China (e-mail: xupengjie194105@sjtu.edu.cn).

Domain Self-Attention mechanism into the data encoding process that enables intelligent vehicles to obtain more delicate environmental clues and perform better navigation. 3) We introduce the Recurrent Neural Network (RNN) into the RL-agent training to enable intelligent vehicles to output the global behavior decision with local observation. 4) We propose a Self-Assessment Gradient Model (SAGM) to evaluate the policy in each state to deal with the suboptimal problem of policy in the process of agent training.

II. Related Work

Autonomous navigation has been extensively researched in the field of vehicles. The existing navigation methods used for vehicles to travel complex and unpredictable environments can be classified as single-modal navigation and multi-modal navigation.

A. Vehicle Navigation based on a Single-modal Perception

Single-modal perception offers certain benefits, such as simplicity, low cost, and ease of implementation. However, it also has drawbacks, including limited information, susceptibility to interference, and difficulty in adapting to complex and changing situations.

Chen et al. [13] proposed a method for obstacle avoidance using a single camera and deep neural network. Rao et al. [14] developed an autonomous navigation system using a single-camera sensor and investigated essential technologies. Liu et al. [15] proposed a vehicle obstacle avoidance method that relies on a single camera and convolutional neural network (CNN). Wang et al. [16] proposed a method for obstacle avoidance using a single camera and DRL.

However, obstacle avoidance based on single-modal vision perception has limitations such as restricted information, susceptibility to interference, and difficulty in adapting to complex and changing environments. In this paper, we propose a multimodal visual perception method for vehicle navigation in complex environments.

B. VehicleNavigation based on a Multi-modal Perception

Multi-modal perception is the process of collecting environmental data using a variety of sensors, offering the advantages of rich information, and strong robustness.

Liu et al. [17] proposed a method for obstacle avoidance based on multi-sensor fusion and DRL, aimed at generating suitable obstacle avoidance strategies. Wang et al. [18] proposed an obstacle avoidance method that utilizes multimodal fusion and a graph neural network (GNN) to effectively identify and localize various types of obstacles outdoors. Xiao et al. [19] proposed a method for autonomous navigation based on DRL and multimodal fusion. Yu et al. [20] proposed a cross-scene place recognition framework that relies on multi-modal deep features.

In complex environments, different sensing modalities may produce different features, which may lead to a degradation in model performance. Currently, researchers are exploring various methods to solve this problem, such as multi-task learning, attention mechanisms, etc. These methods can help models better describe features in complex environments, thereby improving performance. However, this is still an active area of research, and more research is needed to resolve this issue. In our work, we build a cross-attention mechanism based on the attention mechanism and apply it to the multi-modal model to enhance the representation of salient features.

C. Vehicle Navigation based on Policy Learning

DRL-based vehicle navigation is a technique that utilizes reinforcement learning methods to train vehicles to autonomously plan and execute navigation behaviors based on environmental observations and goal instructions, without depending on pre-existing maps.

Akmandor et al. [21] proposed a DRL-based navigation approach that defines the occupancy observations as heuristic evaluations of motion primitives, rather than relying on raw sensor data. Quillen et al. [22] conducted an extensive study on the vision-based vehicle grasping problem, comparing four off-policy deep reinforcement learning algorithms: DQN, DDPG, NAF, and D4PG. Wang et al. [23] introduced a hierarchical deep reinforcement learning framework with notable sampling efficiency and sim-to-real transfer from simulation to real-world scenarios, enabling fast and safe navigation. Yasser et al. [24] proposed a novel multi-modal fusion framework with latent DRL to enhance urban autonomous driving. Li et al. [25] proposed a recurrent deduction deep reinforcement learning model for multimodal vision-vehicle navigation. Wang et al. [26] proposed an enhanced cross-modal matching (RCM) method that enforces cross-modal navigation locally and globally through RL, aiming to guide embodied agents to perform natural tasks in real three-dimensional environments. In addition, DRL also plays an important role in the field of machine intelligence, and its application has an important reference role for the research of this paper. For example, Zhao et al. [27] constructed realistic vehicle operation scenarios using physics-based sound simulations and proposed the Intrinsic Sound Curiosity Module (ISCM), which provides feedback to reinforcement learners to learn robust representations and reward more effective exploratory behaviors. Intelligent vehicles must achieve autonomous steering in various changing navigation environments. Wu et al. [28] proposed a novel end-to-end network architecture to achieve autonomous learning steering of vehicles through DRL. Burhan Hafez et al. [29] proposed to continuously learn control tasks by DRL that performs unsupervised learning of behavioral embeddings by incrementally self-organizing demonstration behaviors.

However, because the decision-making model lacks long-term dependence on data association, it is difficult to achieve global behavioral decisions with local observations. To solve this problem, we introduce the RD model and SAGM into the recurrent neural network (RNN) in the DRL process to obtain the long-term dependencies of decision-making associations.

III. Multi-modal perception-based DRL navigation framework

We aim to introduce a visual perception-based DRL approach with high training efficiency and generalization abil-

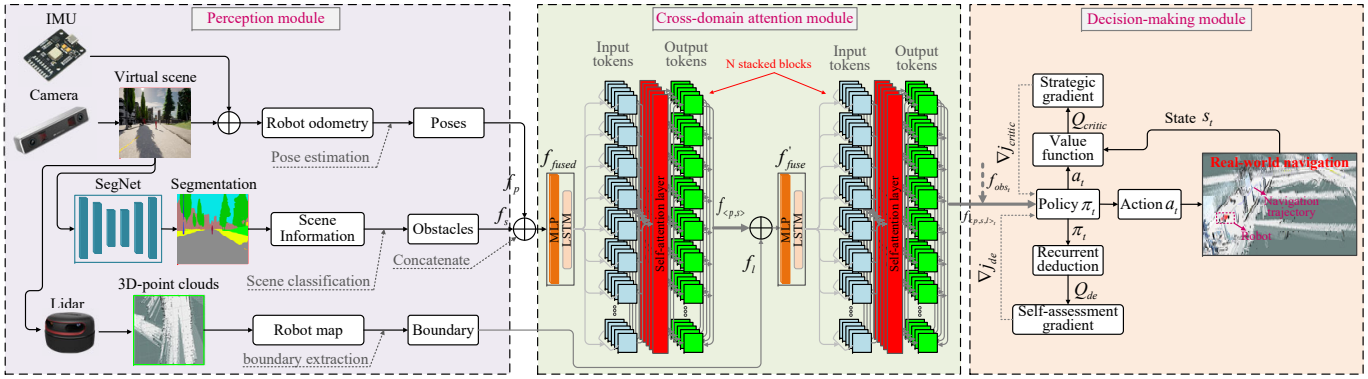

Fig. 1. The visual perception-based DRL navigation framework. The perception module is used to extract visual clues, the cross-domain attention module is responsible for selecting significant and salient information, and the decision-making module is aimed at high-level and safe obstacle avoidance.

ity for fast and safe navigation in complex environments, adaptable to various surroundings and vehicle platforms. To achieve this goal, a three-stage end-to-end vehicle navigation framework, as shown in Fig. 1, is constructed. The proposed DRL framework offers exceptional global decision-making and reasoning capabilities, applicable to diverse environments and various vehicle platforms for the following two reasons: 1) It incorporates a Recurrent Deduction (RD) model based on LSTM, which integrates previous observations into the current decision-making process; 2) It integrates a Self-Assessment Gradient Model (SAGM) into the RD process, evaluating the deductive reasoning of each unit and the overall deduction process based on the overall reward.

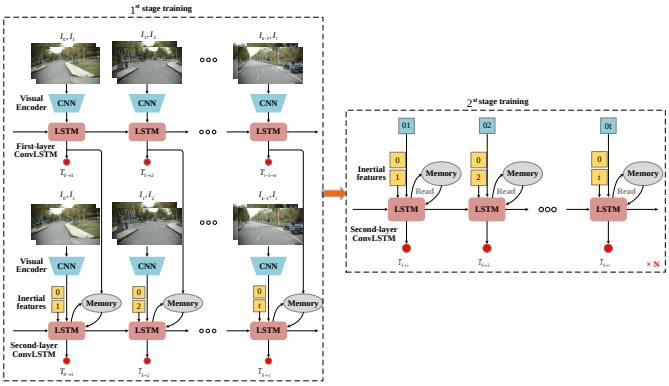

Fig. 2. The overview of the proposed VIO framework. Our method utilizes a stage-wise training approach. In the initial stage, both the first and second layers of the $ConvLSTM$ module are trained simultaneously. In the second stage, only the second layer of the ConvLSTM module is being fine-tuned.

A. Problem Formulation

In obstacle-dense environments, intelligent vehicles require the ability to navigate, making decisions about obstacle avoidance and goal-reaching based on their state and the surrounding information collected by attached sensors. We define a DRL problem as a Partially Observable Markov Decision Process (POMDP) and present it as a tuple (S, A, P, R, Ω, O, γ), wherein S is a set of states with partially observable elements, A is a set of actions of the agent, P is transfer function ($S \times A \to S$), R is the reward function ($S \times A \to R$), $\Omega = \{o\}$ represents a set of observable data o relied on the probability distribution $o \sim O(s)$, and γ represents discount factor ($\gamma \in [0, 1]$).

For each time t, the vehicle agent can recieve observation $o_t \in \Omega$ at current state $s_t \in S$, and output the current action a_t according to the learned policy ($\pi_t = \pi(S_t)$). For next time $t+1$, the vehicle will receive a reward $r(s_t, a_t)$, transition to the next state s_{t+1}, and will receive a new observed information o_{t+1} from the environment. In this process, the reward received by the vehicle from the current state o_t is a kind of long-term cumulative reward $V_\pi(S) = \sum_{k=t}^{T} \gamma^{k-t} R(s_k, a_k)$ under the condition of discount factor γ. The goal of RL is to maximize the reward value obtained from the initial state by optimizing strategies. According to the Bellman Equation (BE), an expected reward can be described as an $Action - Value$ function, i.e. Q value function:

$$Q_\pi(s_t, a_t) = \mathrm{E}_{r_t, s_t \sim E}\left[r(s_t, a_t) + \gamma Q_\pi(s_{t+1}, \pi(s_{t+1}))\right] \tag{1}$$

B. Multi-modal Perception Model

Figure 1 presents a high-level overview of the proposed multi-modal perception-based DRL navigation system. The figure provides a detailed description of the perception module. The odometer, semantic segmentation, and radar perception are the three components of the built-in perception module.

1) Pose estimation: We build a Visual-Inertial Odometry (VIO) system in our work, inspired by [30], to estimate the vehicle's motion trajectory and pose. Our system comprises an end-to-end two-stage pose network consisting of two pose prediction heads, a two-layer LSTM module, and a FlowNet backbone [31]. In the two-layer recurrent architecture, the first layer focuses on predicting the movements of subsequent frames, while the second layer refines the estimates made by the first layer. The proposed VIO framework is shown in Fig. 2.

Given two input data frames (images: I_t, I_{t+1}; imu: $I_t^{'}, I_{t+1}^{'}$), we utilize a recurrent network structure with a

CNN-LSTM ($ConvLSTM$) model to construct a pose estimator. Previously, the pose network directly outputs a 6-degree-of-freedom (6-DoF) pose by concatenating two continuous frames. When predicting the outcome after adding the $ConvLSTM$ module, the pose network also incorporates the data from the previous estimation, which can be described as:

$$F_t = PoseNet\{(I_t, I_{t+1})\} \tag{2}$$

$$O_t, H_t = \text{ConvLSTM}\left(F_t, H_{t-1}\right) \tag{3}$$

$$\hat{T}_{t-1\to t} = L_1(O_t) \tag{4}$$

Where $PoseNet$ is the feature encoder, O_t, H_t present the output and the hidden state of at time t, and $L_1(\cdot)$ is a linear layer to predict the 6-DoF motion $\hat{T}$. By doing this, the network implicitly picks up the ability to combine temporal data and learn motion patterns.

In the sequential modeling setup above, the pose network estimates the relative pose of every two consecutive frames. However, the motion between consecutive frames is often small, making it difficult to extract good features for relative pose estimation. Therefore, self-supervised long-term odometry predicting poses from non-adjacent frames to the current frame (i.e., frame processing across time periods, e.g. 0→T) may be a better choice.

Therefore, we also compute the pose features for the first frame (t=0) and the current visual frame and additional imu frame (t=T) as input to the second-layer $ConvLSTM$:

$$F_t^{'} = PoseNet\{(I_0, I_t), (I_0^{'}, I_t^{'})\} \tag{5}$$

$$O_t^{'}, H_t^{'} = \text{ConvLSTM}\left(F_t^{'}, M_t, H_{t-1}^{'}\right) \tag{6}$$

$$\hat{T}_{t-1\to t}^{'} = L_2(O_t^{'}) \tag{7}$$

Where $O_t^{'}$, $H_t^{'}$ present the output and the hidden state of the $ConvLSTM$ from the second layer at time t, M_t is the read-out memory, and $L_2(\cdot)$ is a another linear layer to predict the 6-DoF motion $\hat{T}^{'}$.

To train the second layer ConvLSTM, we utilize the photometric error X_p between the first frame and the other frames of the input snippet to achieve cycle consistency of the poses of the two layers. It should be noted that we calculate the weighted average of all memory slots in the memory buffer when reading data from it. We also have an additional constraint to ensure consistency between the first and second layers, based on the transferability of sensor transformations.

$$X_P = \frac{1}{N-1}\sum_{t=1}^{N-1}\left\|\hat{T}_{0\to t} - \left(\hat{T}_{t-1\to t}\hat{T}_{0\to t-1}\right)\right\|_2^2 \tag{8}$$

where N is the number of frames for the input snippet, which is set as 5 in our work.

After completing the first stage of training, we run this model separately on each sequence in the dataset to extract and store the input $ConvLSTM$ required for the second layer. After that, we only fine-tuned the lightweight $ConvLSTM$ in the second layer, eliminating the need for a deep network and labor-intensive feature extraction. As a result, we can now train the network using longer sequences, which helps it become more proficient at utilizing temporal context. Therefore, the training loss for the entire stage can be described as:

$$L_{loss} = \frac{1}{M}\sum_{m=0}^{M-1}\frac{1}{N-1}\sum_{t=m(N-1)+1}^{m(N-1)+N-1}\chi\left(I_{m(N-1)}, \hat{I}_{t\to m(N-1)}\right) \tag{9}$$

Where N is the number of frames of each input group, M is the number of groups of each input sequence, $\hat{I}_{t\to m(N-1)}$ represents a function of pose that encodes long-range constraints by the use of $ConvLSTM$.

To facilitate the expression of the subsequent calculation process, we describe the output pose of the vehicle at time t as:

$$f_{p_t} = \hat{T}_{t-1\to t}^{'} \tag{10}$$

2) Semantic Segmentation: We train a lightweight segmentation model (improved based on ENet [32]) to segment images into semantic parts with predicted high-quality pose configurations f_{p_t}. Except for the initial score map, we defined two pose feature maps from f_{p_t}: a joint label map and a skeleton label map, and used them as input segmentation networks for the semantic segmentation thread. We joint the 2-D feature map of the estimated pose with the original segmentation fractional map to generate multi-dimensional inputs and stack three additional convolutional layers. The kernel size is 7, the kernel dimension is 128, and Relu is the Activation function. Then use the Argmax value in the output score map to derive our final segmentation result f_s that is consistent in time with the input frame in the pose estimation.

3) LiDAR Perception: We first preprocess the received data $\mathbf{P}$ before feeding it to the network. We remove the ground points from the point cloud $\mathbf{P}$ and then convert it into a voxel grid with a resolution of 72 × 72 × 48 voxels along the X, Y, and Z dimensions respectively.

We keep track of the planes and lines in the respective candidate sets C_p and C_l over time. Similar to feature tracking techniques, which involve tracking features in the immediate region around their expected location. The tracking planes and lines from the previous scan are first obtained. To facilitate local tracking, we segment C_p and C_l around the expected feature position by utilizing the maximum point-to-model distance. After that, we remove outliers using Euclidean clustering and standard plane feature filtering. Finally, we utilize the robust fitting algorithm (PROSAC) [33] to fit the model to the segmented point cloud. Finally, we need to check if the predicted and detected boundaries are sufficiently similar. Two planes, p_i and p_j, are considered matched when the difference between their unit normal vectors $\hat{n}_i$, $\hat{n}_j$ and the distance from the origin are smaller than a threshold [34].

We define that two boundaries l_i and l_j are considered to match if their direction and center distance are less than the following threshold:

$$\eta_n^{'} = \|\arccos(\hat{v}_i \cdot \hat{v}_j)\| < \alpha_l \tag{11}$$

$$\eta_d^{'} = \|(d_i - d_j) - ((d_i - d_j)\cdot\hat{v}_i)\hat{v}_i\| < \beta_l \tag{12}$$

Where $\hat{v}$ represents the normal of a boundary, and $\alpha_p = \alpha_l = 0.38rad$, $\beta_p = \beta_l = 0.55m$.

The process is repeated for the remaining boundaries after

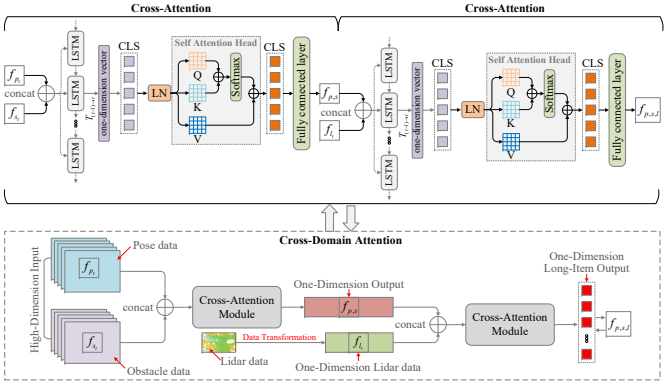

Fig. 3. The Overview of the Proposed Cross-domain Attention Model. Where CLS (class token) is a special token that is used to encode original image patches, LN (layer normalization) is a normalization method.

a feature has been tracked, removing its inner line from the corresponding candidate set. After completing the tracking, we identify new boundaries in the remaining candidate clouds. The line is initially segmented into point clouds using Euclidean clustering, and the plane is segmented into point clouds using region growth based on normals. Next, to determine new boundaries within each cluster, we use the same technique as boundary tracking. Finally, the boundary features extracted from point clouds are represented as:

$$f_l = Euclid\{(\eta_n, \eta_d), (\eta_n^{'}, \eta_d^{'})\} \tag{13}$$

C. Cross-domain Attention Model (CDSAttention)

The CDSAttention consists of two identical cross-attention modules. The first module focuses on integrating pose information and semantic scene information, while the second module addresses the further fusion of the output of both fusions with LiDAR scanning. The overview of the proposed CDSAttention model is shown in Fig. 3.

We aim to utilize cross-attention modules to integrate pose representations and semantic representations within the scene. The cross-attention module is trainable, and it can be intuitively understood that the learned fusion strategy is superior to the manually crafted orthogonal fusion strategy.The CDSAttention is constructed by stacking M cross-attention functions $\mathcal{T} : (\mathbb{R}^{1\times C_1}, \mathbb{R}^{1\times C_2}) \rightarrow \mathbb{R}^{1\times C}$ that integrates the two short one-dimensional vector of pose f_p and vector of segmentation information f_s into a long one-dimensional fused vector of output $f_{<p,s>}$, wherein the C_1, C_2, and C represent length of encoded binary data. Formally, we encode pose data and segmentation to be $f_p \in \mathbb{R}^{1\times C_1}$ and $f_s \in \mathbb{R}^{1\times C_2}$, and the final one-dimensional descriptor $f_{p,s}$ can be represented by:

$$f^m_{<p,s>} = \mathcal{T}_m(f_p^{(m-1)}, f_s^{(m-1)}), m \in \{1, 2, ..., M\} \tag{14}$$

The cross-attention function is illustrated in detail in Fig. 3 and can be calculated as:

$$\begin{aligned} &\mathcal{T}_m\left(f_p^{(m-1)}, f_s^{(m-1)}\right) = \\ &FFN_m\left(MCA_m\left(f_p^{(m-1)}, f_s^{(m-1)}\right) \| f_p^{(m-1)}\right) + f_p^{(m-1)} \end{aligned} \tag{15}$$

Where $FFN_m(.)$ represents the training of a two-layer feed-forward network, MCA_m is a multi-head attention module that aggregates information between pose data f_p and segmentation data f_s, and $\|$ represents concatenation process. Specifically, f_p and f_s are first processed by LSTM for spatiotemporal consistency and then fused by $MCA_m(f_p^{(m-1)}, f_s^{(m-1)}) = (CA_m^1\|, ..., \|CA_m^{(n_h)})W^{(m)}$, where CA_m^i represents the i^{th} output from cross-attention module, n_h denotes the number of attention heads, and $W^{(m)}$ represents projection matrix. Regarding the calculation method of self-attention, CA_m can be calculated by the follows:

$$CA_m = \text{Softmax}\left(\frac{fp^{(m-1)}W_Q^{(m)}(f_sW_K^{(m)})^T}{\sqrt{C/n_h}}\right) f_sW_V^{(m)} \tag{16}$$

Where $fp^{(m-1)}W_Q^{(m)}$ is the query, $f_sW_K^{(m)}$ is the key, and $f_sW_V^{(m)}$ is the value projection matrix, respectively.

Similar to the first cross-attention process, we continue to perform the second cross-attention to fuse $f_{<p,s>}$ with f_l. We first describe the second cross-attention function as: $\mathcal{T}^{'} : (\mathbb{R}^{1\times C}, \mathbb{R}^{1\times C_3}) \rightarrow \mathbb{R}^{1\times C^{'}}$ that integrates the one-dimensional vector of pose-segmentation $f_{<p,s>}$ and vector of Lidar information f_l into a long one-dimensional fused vector of output $f_{<p,s,l>}$, wherein the C, C_3, and $C^{'}$ represent length of encoded binary data. Similarly, we also encode pose-segmentation data and Lidar to be $f_{<p,s>} \in \mathbb{R}^{1\times C}$ and $f_l \in \mathbb{R}^{1\times C_3}$, and the final one-dimensional descriptor $f_{<p,s,l>}$ can be represented by:

$$f^m_{<p,s,l>} = \mathcal{T}^{'}_m(f_{<p,s>}^{(m-1)}, f_l^{(m-1)}), m \in \{1, 2, ..., M\} \tag{17}$$

The second cross-attention function $\mathcal{T}^{'}$ can be calculated as:

$$\begin{aligned} &\mathcal{T}^{'}_m\left(f_{<p,s>}^{(m-1)}, f_l^{(m-1)}\right) = \\ &FFN^{'}_m\left(MCA^{'}_m\left(f_{<p,s>}^{(m-1)}, f_l^{(m-1)}\right) \| f_{<p,s>}^{(m-1)}\right) + f_{<p,s>}^{(m-1)} \end{aligned} \tag{18}$$

Finally, the $CA^{'}_m$ in the process of the second cross-attention module can be calculated as:

$$CA^{'}_m = \text{Softmax}\left(\frac{f<p,s>^{(m-1)}W_Q^{(m)^{'}}(f_lW_K^{(m)^{'}})^T}{\sqrt{C/n_h}}\right) f_lW_V^{(m)^{'}} \tag{19}$$

Where $f<p,s>^{(m-1)}W_Q^{(m)^{'}}$ is the query, $f_lW_K^{(m)^{'}}$ is the key, and $f_lW_V^{(m)^{'}}$ is the value projection matrix in the second cross-attention process, respectively.

D. Decision-making Model

Deductive reasoning enables the agent to predict future driving trajectories by learning the environmental model. Since the simulated environment is used for policy learning and the learned policy is applied to the real environment, the environment model is obtained by training on tuples (o_t, a_t, r_t, o_{t+1}) sampled from replay experiences..

The agent processes the environmental parameters input by the perception module and continuously updates the action-environment state through recurrent deductive reasoning (RDR). The LSTM is integrated into the agent's strategy learning for global behavior decision-making in motion planning for vehicles. It continuously inputs local observation data into the network and updates the environment states to perform global behavior decisions in the local environment, which is outlined in Fig. 4.

Constructing a transition model to move from the current

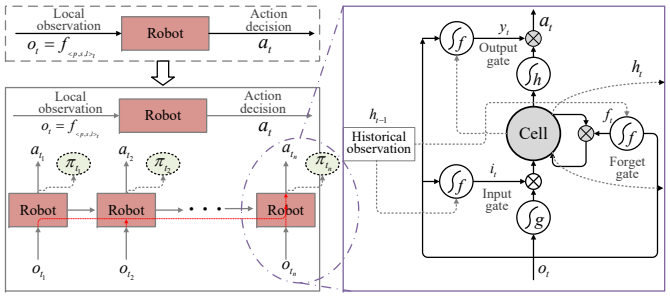

Fig. 4. The overview of the proposed deductive reasoning-based decision-making model.

state a_t to the next state a_{t+1} is necessary when learning the environment model $o_t = (f_{<p,s,l>_t}, o_{obs_t})$ to update the state and strategy. The transition model first takes in the current observations o_t and action a_t before making predictions about the environment state o_{t+1} at the following instant. The transition model is characterized as:

$$\tilde{o}_{t+1} = T\left(o_t, a_t\right) \tag{20}$$

Where $\tilde{o}_{t+1}$ is the state of the next moment, T is the state transition function. The loss function can be expressed as:

$$L_\xi\left(\theta^\xi\right) = \left\| r_t - \xi\left(o_t, a_t, o_{t+1}\right)\right\|^2 \tag{21}$$

Where θ^ξ is the strategic function, r_t is the obtained reward at time t. The reward value $\xi(o_t, a_t, \tilde{o}_{t+1})$ of the moment t can be obtained by the reward function $\xi(.)$.

Then, the RDR model based on LSTM imports the currently learned strategy, the environment state, and the reward obtained based on the strategy into the LSTM model to obtain the output at the next moment. The current observation and history observation of the input of the LSTM model input gate can be expressed as $x_t = (\tilde{o}_t, r_t)$ and $h_{t-1} = (\tilde{o}_{t-1}, r_{t-1})$.

LSTM-based DR achieves the selective transfer of historical memory by incorporating the cell state into the recurrent unit, distinguishing them from traditional recurrent neural networks (RNNs). Updates to memory cells are necessary during this process:

$$c_t = f_t \cdot c_{t-1} + i_t \cdot (\mathrm{Tanh}\left(w_c\left(\tilde{o}_t, r_t\right) + u_c\left(\tilde{o}_{t-1}, r_{t-1}\right)\right)) \tag{22}$$

Where $Tanh(\cdot)$ is a nonlinearized activation function, w_c and u_c are the weight matrix at time t and time $t-1$.

The purpose of updating the memory unit is to utilize the selected memory information c_t as a "mask" to derive the output state. According to the DR of LSTM in Fig. 4, the output of the final unit is determined by the input. The value of going out is combined with the currently selected memory information to obtain:

$$h_t = y_t \cdot \mathrm{Tanh}\left(\mathrm{Tanh}\left(w_c\left(\tilde{o}_t, r_t\right) + u_c\left(o_{t-1}, r_{t-1}\right)\right)\right) \tag{23}$$

While LSTM-based DR can make global decisions during state updates, the complexity and variability of the environment, as well as the unpredictability of the vehicle's state, may lead to inaccurate reasoning at specific points. This, in turn, can impact the task of motion planning for vehicles. To address this issue, we construct a SAGM to implement decision-making corrections. When the vehicle receives an observation o_t at time t, the DR model learns a strategy π_t and predicts the vehicle's behavior a_t based on the strategy π_t. Then, the environment model will predict the next observation $\tilde{o}_{t+1}$, obtain a reward r_t at the same time of prediction, and then obtain a reward r_{t_n} at the n^{th} time step t_n after RDR.

SAGM evaluates the performance of the entire inference process by integrating rewards over multiple time steps. The total reward $Q_{SAGM}(o_t)$ represents the evaluation value at the current observation state, with a discount factor β used to balance short-term and long-term rewards:

$$Q_{SAGM}(o_t) = \sum_{k=1}^{n} \beta^{k-1} r_{t+k-1} + \beta^n V(\tilde{o}_{t+n}) \tag{24}$$

Where $V(\tilde{o}_{t+k})$ is the state value under environment state $\tilde{o}_{t+k}$, β is the discount factor.

In policy learning, the model uses an "experience replay" strategy, optimizing the policy parameters using historical samples. The Q-function is updated by minimizing the Bellman loss, where the target value y_t depends on experience replay samples:

$$L(\theta^Q) = E_{o_t, a_t, r_t, s_{t+1} \sim B}(y_t - Q(s_t, a_t|\theta^Q)) \tag{25}$$

Where y_t is the target value. The $Actor$ network learns an optimized driving policy and optimizes the policy parameters using sampled policy gradients:

$$\begin{aligned} \bigtriangledown_{\theta_\pi} J \approx E_{o_t, a_t, r_t, s_{t+1} \sim B}[\bigtriangledown_a Q(s, a|\theta^Q)|_{s=s_t, a=\pi(s_t)} \\ z \bigtriangledown_{\theta_\pi} \pi(s|\theta_\pi)|_{s=s_t}] \end{aligned} \tag{26}$$

To make the training process more stable and reliable, a target $Actor$ network and a target $Critic$ network are introduced, and the target network is updated with the strategy learning:

$$\theta^{'} \leftarrow \tau\theta + (1-\tau)\theta^{'} \tag{27}$$

Where τ is the update rate of the network.

The state evaluation from the $Critic$ network is combined with the self-assessment results from SAGM through weighted summation, producing a combined Q-value:

$$Q_{total}(s_t, a_t) = Q_{critic}(s_t, a_t) + \gamma Q_{SAGM}(o_t) \tag{28}$$

Where γ is the weight parameter of the SAGM. Finally, The strategy will be further optimized through the strategy gradient model:

$$\bigtriangledown_{\theta^\pi} J = \bigtriangledown_{\theta^\pi} J_{critic} + \omega \bigtriangledown_{\theta^\pi} J_{SAGM} \tag{29}$$

Where J_{critic} and J_{SAGM} represent the gradient learned from the $Critic$ network and the SAGM, respectively.

SAGM refines autonomous decision-making by combining immediate observations with long-term outcome evaluations. At each step, the model predicts future states, assesses short-term rewards, and calculates cumulative value using a discount factor to balance immediate and future rewards. Through experience replay, SAGM optimizes Critic network Q-values and Actor network policy gradients, then integrates self-assessment feedback with Critic evaluations to produce a combined Q-value. This approach enables adaptive policy adjustments,

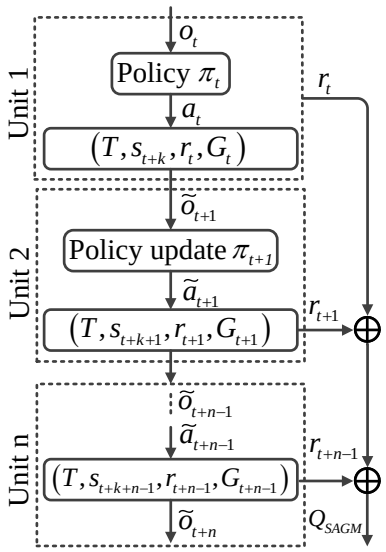

Fig. 5. Policy optimization based on the Self-assessment gradient model (SAGM).

allowing for effective self-correction and enhanced stability in dynamic environments. The complete process is detailed in Fig. 5. It should be noted that the environment model (T, s_{t+k}, r_t, G_t) mainly includes state transition functions and reward functions.

IV. Experimental verification

We carry out experiments on vehicle navigation in both simulated and real-world environments to demonstrate the superiority of the proposed model over the baseline methods.

A. Experiment Setting

The proposed navigation framework is an end-to-end trainable model comprising a perception module with three parallel sub-perception threads and a decision-making module. We train and test our automated vehicle using a CARLA simulator, which offers a high-fidelity driving environment with diverse traffic conditions [35]. For a fair comparison with other baseline methods, we train the proposed model and baseline models in the same scenes, following identical environment settings. Extreme weather conditions in the training set include rainy days, rainy nights, foggy days, and cloudy days, while clear nights and clear days are not utilized in the training stage. The proposed navigation model is trained for 15k episodes with a batch size of 128 and all images are resized to 64×64. The model is optimized using an ADAM optimizer with an initial learning rate of 0.0001. The I7-12700 CPU and 12G 3060 GPU of the Ubuntu 18.04 hardware platform are used to run the entire simulated experiment.

In our work, the multi-modal perception model and the sensors on the simulation platform both contribute to one of the two components that make up the state observation ($o_t = (o_{s_t}, o_{obs_t})$) of RL. The measurement as the part-state o_{s_t}=$(v_{f_t}, v_{l_t}, \varphi_t, d_{0_t}, d_{1_t}, d_{2_t}, d_{3_t}, d_{4_t})$ is obtained using the sensor from the CARLA simulator, where v_f and v_l represent forward speed and lateral speed, φ is the related angle, d_0, d_1, d_3, and d_4 represent road width, the distance to lane, the distance to other vehicles, and the distance to pedestrian, respectively, $d_{2_t} = d_{0_t}/2 - d_{1_t}$. The observation from environment as another part-state $o_{obs_t} = f_{<p,s,l>_t} = (f_{p_t}, f_{s_t}, f_{l_t})$ is obtained using the proposed multi-modal cross-domain model. Therefore, the explicit form of the reward function $\tilde{r} = \xi(.)$ in equations (23) and (24) can be defined as the sum of the above six terms:

$$\tilde{r} = r_{v_{f_t}} + r_{v_{l_t}} + r_{\varphi_t} + r_{d_{2_t}} + r_{d_{3_t}} + r_{d_{4_t}} \tag{30}$$

Considering the importance of driving safety, the model is trained by setting reward conditions in automatic vehicle navigation:
Conditions 1 & 2:

$$r_{v_{f_t}} = \begin{cases} v_{f_t}, v_{f_t} \leq 30 \\ 60 - v_{f_t}, v_{f_t} \geq 30 \end{cases} \quad r_{v_{l_t}} = \begin{cases} v_{l_t}, v_{l_t} \leq 10 \\ 10 - v_{l_t}, v_{l_t} \geq 10 \end{cases} \tag{31}$$

Conditions 3:

$$r_{\varphi_t} = \begin{cases} r_{\varphi_t}, r_{\varphi_t} \leq 90 \\ 180 - r_{\varphi_t}, r_{\varphi_t} \geq 90 \end{cases} \tag{32}$$

Conditions 4 & 5 & 6:

$$r_{d_{2_t}} = \begin{cases} 0, d_{2_t} \geq d_{0_t}/2 \\ d_{2_t}, d_{2_t} \leq d_{0_t}/2 \end{cases} \tag{33}$$

Conditions 5 & 6:

$$r_{d_{3_t}/d_{4_t}} = \begin{cases} 0, \leq d_{3_t}/d_{4_t} \geq 0.2 \\ d_{3_t}/d_{4_t}, d_{3_t}/d_{4_t} > 0.2 \end{cases} \tag{34}$$

During training and inference, the robot has a set of available actions (Action Set) at each step, including: Forward, Backward, Turn Left, Turn Right, emergency halts. Each directional action can be executed at different speed levels (e.g., low, medium, high) for fine-tuning in various situations.

For continuous control, the action space A can be represented as a multi-dimensional vector, where each dimension corresponds to a control variable, such as speed v and θ:

$$A = \{(v, \theta) \mid v \in [v_{\min}, v_{\max}], \theta \in [\theta_{\min}, \theta_{\max}]\} \tag{35}$$

During inference, the policy function, π selects the optimal action a_t based on the robot's current state s (including multi-modal perception data).

During training, we optimize the policy network using DDPG with SAGM optimization. The goal of the policy network is to maximize the expected cumulative reward: G_t, which represents the total expected reward from the initial state s_0:

$$G_t = \mathbb{E}\left[\sum_{k=t}^{T} \gamma^{k-t} \tilde{r}_k\right] \tag{36}$$

Where γ is the discount factor used to balance short-term and long-term rewards.

This reward design effectively directs the model to learn obstacle avoidance while keeping the navigation goal-oriented.

B. Model Compression and Parameter Freezing in Sim-to-Real Transfer

To enable efficient transfer of the intelligent vehicle navigation model from simulation to real-world environments, we employed model compression and parameter freezing techniques to enhance computational efficiency and stability.

1) Model Compression:

- Weight Pruning: reducing model size by removing redundant parameters from the network. Specifically, we define the weight matrix of each network layer as $W \in R^{n\times m}$, where n and m are the input and output dimensions of the layer, respectively. We use a threshold ϵ to prune weights, setting those below the threshold to zero:

$$W_{i,j} = \begin{cases} W_{i,j} & \text{if } |W_{i,j}| \geq \epsilon \\ 0 & \text{if } |W_{i,j}| < \epsilon \end{cases} \tag{37}$$

  Where $W_{i,j}$ is the element of the weight matrix W at row i and column j, representing the weight value for that specific connection. By optimizing pruning threshold ϵ, we effectively reduce the number of parameters with minimal impact on performance. The pruned network undergoes fine-tuning to recover any lost performance.
- Quantization: Model quantization converts floating-point parameters to lower-precision integers, thus reducing memory usage and computational complexity. We use 8-bit quantization, converting 32-bit floats to 8-bit integers. Assuming a weight range of $[W_{\max}, W_{\min}]$, each weight W is quantized as:

$$W_q = \text{round}\left(\frac{W - W_{\min}}{W_{\max} - W_{\min}} \times \left(2^8 - 1\right)\right) \tag{38}$$

  Where W_q is the quantized weight value, $(2^8 - 1))$ represents the range for 8-bit quantization, from 0 to 255, round$(\cdot)$ represents the rounding operation. Specifically, round$(\cdot)$ rounds a floating-point number to the nearest integer.
- Knowledge Distillation: Transferring knowledge from a large pre-trained teacher model T to a smaller student model S, For an input sample x, the loss between the teacher's output y_T and the student's output y_S is defined as:

$$L_{\text{distill}} = \alpha \cdot L_{\text{CE}}(y_S, y_{\text{true}}) + (1-\alpha) \cdot L_{\text{KL}}(y_S, y_T) \tag{39}$$

  Where L_{CE} is the cross-entropy loss, L_{KL} is the Kullback-Leibler divergence, and α is a hyperparameter balancing the two. Through knowledge distillation, the student model efficiently learns the teacher model's representational capacity, maintaining performance while reducing parameters.

2) Parameter Freezing:

- Layer-wise Freezing: In this study, we freeze the lower-level feature extraction layers (e.g., convolutional layers) in the perception module. Let the network's layer weights be $\theta = \{\theta_1, \theta_2, \ldots, \theta_L\}$, with the first k layers layers frozen, $i.e.\{\theta_1, \theta_2, ..., \theta_k\}$. The parameter update rule for frozen layers is:

$$\theta_i^{'} = \theta_i, \quad \forall i \leq k \tag{40}$$

- CDSAttention Module Freezing: The Cross-Domain Self-Attention (CDSAttention) module plays a key role in fusing multi-modal data. We freeze this module to maintain its robustness in integrating environmental cues. After freezing, the update rules for weight matrices within the module (e.g., query, key, and value matrices) are:

$$W_Q^{'} = W_Q, \quad W_K^{'} = W_K, \quad W_V^{'} = W_V \tag{41}$$

- Partial Freezing of RD and SAGM: To enhance stability in navigation decisions, we partially freeze key memory units in the Recurrent Deduction (RD) module and the policy evaluation part of the Self-Assessment Gradient Model (SAGM). Let the memory state in the RD module be h_t; after partial freezing, its update rule becomes $h_t = f(h_{t-1}, x_t)$ (partial freezing). For the evaluation function Q_{SAGM} in SAGM, the policy update rule under freezing is:

$$Q_{\text{SAGM}}^{'}(s, a) = Q_{\text{SAGM}}(s, a), \quad \forall (s, a) \in \mathcal{S} \times \mathcal{A} \tag{42}$$

  Where s and a representing the current environment state and decision action in the navigation task, x_t is the observations from the perception module at time t, $\mathcal{S}$ and $\mathcal{A}$ are the state space and the action space, containing all possible combinations of states and actions.

C. Experiments based on Simulated Environments

We conduct experiments using the CARLA simulator to assess the effectiveness of the proposed end-to-end and perception-decision models, employing simulation-to-reality transfer learning technology. CARLA, like many other autonomous driving simulators, is model-based, in contrast to our data-driven simulators. Despite great efforts to make the CARLA environments as realistic as possible, there are still simulation gaps. We found that end-to-end models trained only in CARLA do not perform well in the real world. Therefore, we utilize a highly challenging environment as a virtual training scenario to tackle this issue. Additionally, we incorporate a SAGM and a DR mechanism into the RL model to bridge the gap between the virtual and real environments through the transfer learning technology.

Figure 6 displays examples of scene perception generated by the proposed multi-modal cross-domain module while navigating the vehicle in the CARLA simulator. During the automated driving experiment, we simulated a real environment by randomly introducing moving vehicles, pedestrians, dynamic obstacles, and traffic lights. This allowed the trained agent model to be applicable in real-world scenarios without differentiation.

1) Results Analysis of Agent Training: Figure 7 illustrates the comparison of the training performance between the proposed RD+SAGM and RD without SAGM. As the number of training frames increases, the training strategy using RD+SAGM is more efficient than using RD alone. This results

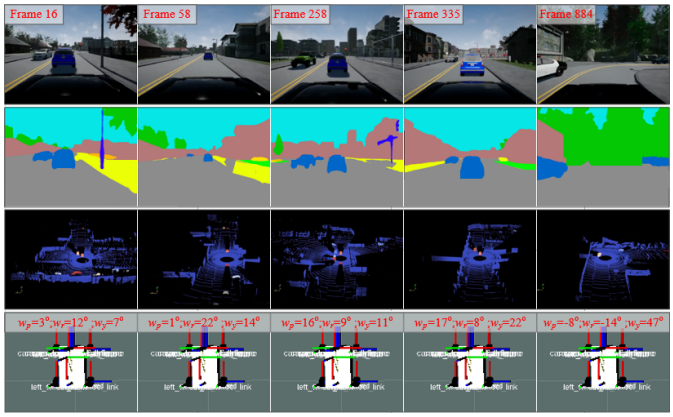

Fig. 6. Perception results are generated by the proposed multi-modal cross-domain module based on the CARLA simulator. From top to bottom, the images show the original, semantic segmentation, LADIR perception, and pose estimation results.

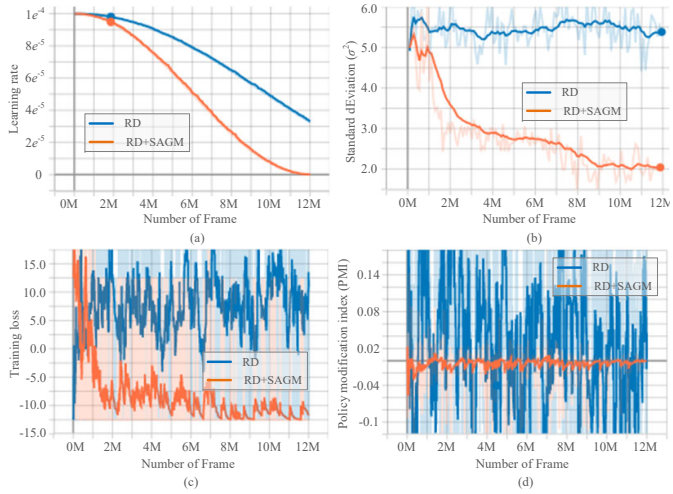

Fig. 7. Comparison of the training performance between the proposed RD+SAGM and RD without SAGM.

in a faster gradient decline of the learning rate, as shown in Fig. 7(a). We use the standard deviation (σ^2) of rewards obtained by the agent to assess the impact of the SAGM on the overall architecture. It can be seen from Fig. 7(b) that the training effect using SAGM is superior. According to the gradient descent of the training loss under the two modes in Fig. 7(c), it can be observed that the training loss decreases more rapidly when using SAGM, suggesting a more effective training outcome. In addition, we utilize the Policy Modification Index (PMI) to further investigate the real-time impact of SAGM on the training process of agents. As shown in Fig. 6(d), the fluctuation of PMI with the SAGM is smaller, indicating a better effect compared to the one without the SAGM.

2) Comparative Results with the State-of-the-art RL Methods: We conduct comparative experiments to validate the effectiveness of the proposed method. We employ state-of-the-art RL methods, including TD3, A3C, and A2C, commonly used in autonomous navigation and decision-making tasks as evaluation baselines. We select these baseline models for their

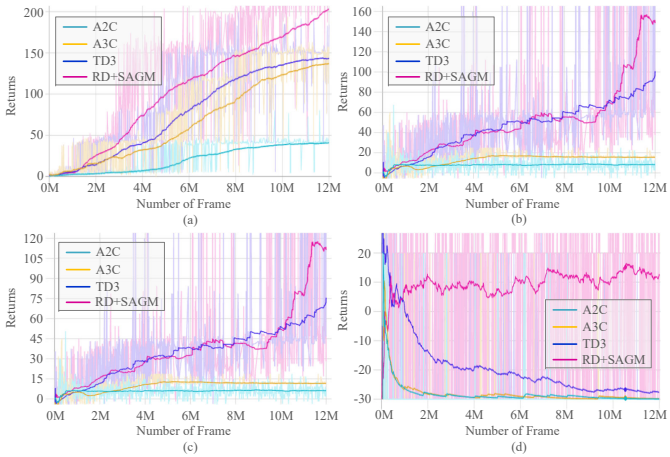

Fig. 8. Comparison of the proposed RD+SAGM and state-of-the-art methods such as Twin Delayed Deep Deterministic Policy Gradient (TD3), Asynchronous Advantage Actor-Critic (A3C), and Advantage Actor-Critic (A2C). (a)-(d): rainy day, cloudy day, rainy night, and foggy day.

maturity and performance in continuous and discrete action spaces, establishing a standard for comparing our multimodal navigation system. This comparison clarifies the advantages of our proposed approach under different conditions. TD3 enhances DDPG with a dual Q-network structure to reduce overestimation bias. A3C is an asynchronous policy gradient method that accelerates training through multi-threading. A2C, the synchronous version of A3C, uses single-threaded batch updates and includes both an Actor and a Critic network. Training results from Fig. 8 under four different weather conditions show that the proposed agent training strategy based on RD+SAGM outperforms other models. In addition, when comparing the training results in different environments, it can be observed that the more severe the weather, the lower the reward degree (returns) of the agent, indicating a poorer training effect. (average returns: $rainy\ day > cloudy\ day > rainy\ night > foggy\ day$).

3) Importance Analysis of Key Component: The impact of vehicle training will vary depending on the perception methods used. To assess the significance of various perception components within the overall framework, we conducted comparative experiments. The experimental results show that the proposed multi-modal perception method (Pos. + Seg. + Lid.) outperforms other methods (Seg., pos. + Seg., and Pos. + Lid.), which is shown in Table I. In addition, when comparing the results of single-modal perception and two-modal perception, it can be observed that the training effect of the agent is based on two-modal perception (pos. + Seg. and Pos. + Lid.) is better than that of single-modal perception (Seg.). Therefore, we can conclude that increasing environmental cues can enhance agents' effectiveness in RL. The rationale behind this is that when a vehicle navigates in a challenging environment and one of its sensors malfunctions, it can use data from other sensors to compensate for the gaps.

4) Stability Evaluation of Autonomous Driving: Furthermore, we utilize the degree of dispersion of the driving position to assess the driving stability of the trained agent. The deviation (standard deviation δ^2) between the vehicle's

TABLE I
AVERAGE REWARD OF THE VARIOUS PERCEPTION COMPONENTS, WHERE SEG., POS. + SEG., POS. + LID. AND POS. + SEG. + LID. RESPECTIVELY, THESE INDICATE THAT THE PERCEPTION MODULE USES ONLY SEGMENTATION INFORMATION, POSE + SEGMENTATION INFORMATION, POSE + LIDAR INFORMATION, AND POSE + SEGMENTATION + LIDAR INFORMATION.

Methods	Weather conditions			
	Rainy day	Cloudy day	Rainy night	Foggy day
Seg.	89.44	69.54	54.76	4.87
Pos. + Seg.	124.39	82.31	62.90	6.07
Pos. + Lid.	130.80	94.66	66.47	6.54
Pos.+ Seg. + Lid.	**133.56**	**98.75**	**73.82**	**8.86**

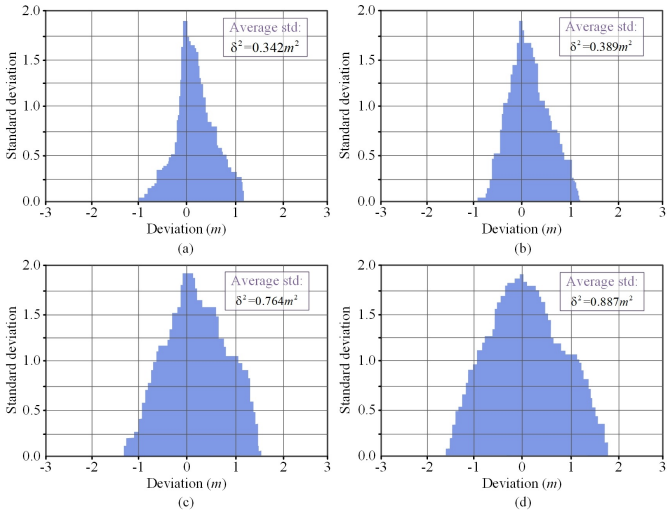

Fig. 9. Degree of off-center driving position. (a)-(d) represents four training scenes: rainy day, cloudy day, rainy night, and foggy day.

position and the center driving position of the road is a metric of the driving stability. By comparing the average standard deviation (average std) of the agent in the four scenarios, it is evident that the complexity of the environment significantly impacts the driving stability of the trained agent. The results in Fig. 9 indicate that training agents on foggy and rainy nights are more challenging than on rainy and cloudy days.

Although both RD models such as A2C, A3C, TD3, and the proposed RD+SAGM can achieve normalized rewards by agent training, the policies they learned are very different from each other. From Table II, we can see that the success rate of the RD+SAGM agent in each of the two environments (day & $night$) is higher than 90%, and achieves 95.7% by average, around $10\%-20\%$ higher than other RD methods. Besides, the collision rate and off-lane rate of the RD+SAGM agent also are far below other RD methods. Therefore, although both four RD methods are designed to address partially observable Markov decision processes (POMDPs), our RD+SAGM is more efficient.

5) Ablation Study: We conduct ablation experiments by varying the number of stacking cross-domain attention blocks in the Transformer to investigate their impact on navigation performance. (a) and (b) in Fig. 10 summarize the results. We found that increasing the number of stacks can further improve the navigation performance of the vehicle in two main ways: 1) enabling the vehicle to find a more effective path to the destination while avoiding collisions (resulting in a shorter driving path); 2) enabling the vehicle to minimize violations and improve the success rate of reaching the destination.

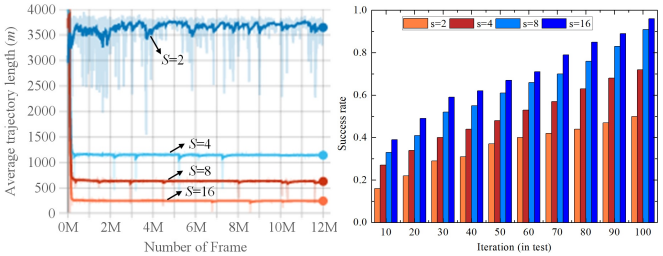

Fig. 10. Experimental results with various stacking cross-domain attention blocks. (a): Average trajectory length with various stacking cross-domain attention blocks. (b): The success rate of autonomous driving using various stacking cross-domain attention blocks.

6) Comparison with SOTA Navigation Approaches: To validate the effectiveness and general applicability in various scenarios, we perform vehicle navigation tests in six scenarios with dynamic obstacles using the CARLA platform. We also compare the approach with existing SOTAs such as manual control [34], DQN-Waypoints [35], DQN-CNN [36], and OGM [37] trained under the same conditions. As shown in Fig. 11, six routes have been established in CARLA's six maps. The pre-trained model follows these routes and records the trajectory while navigating. All agents can follow the path correctly, and all of them completed the designated route. However, the outcome is mixed due to collisions along the way. Table III presents the quantitative results of agents trained using different models. The proposed multi-modal attention perception model (presented in $RD+SAGM$) achieves better results in terms of $RMSE$ and $navigation\ time$. It should be noted that in the navigation tests, the proposed approach shows greater improvement than the "DDPG + RD"-only and "DDPG + SAGM"-only models, respectively. however, "A3C/TD3 + RD + SAGM" schema also present strong performance, as shown in Table III ($\uparrow$ x, y, where x is the improvement compared with RD, y represents the improvement compared with SAGM).

V. TESTS ON REAL-WORLD ENVIRONMENTS

In this section, we provide physical vehicle tests to verify the effectiveness and practicality of our approach in real-world scenarios. As shown in Fig. 11, our experiments consider large-scale outdoor environments and illumination conditions (including day and night tests), which is also the most representative application scenario of vehicles in practice. Including static obstacles (static vehicles, artificial roadblocks, etc.) and dynamic objects (pedestrians, vehicles) existing in the environment. The vehicle is equipped with 3D LiDAR, cameras, a low-level motion control system, and Jetson Xavier NX for model calculations. To implement the application of large models on hardware platforms, we use transfer learning technology to successfully apply the trained model to the physical vehicle platform through model compression and parameter freezing. During the model transfer process, we use

TABLE II
STABILITY EVALUATION OF A2C, A3C, TD3, AND THE PROPOSED RD+SAGM IN DIFFERENT TEST ENVIRONMENTS.

Environments	Success rate (%)				Collision rate (%)				Off-lane rate (%)			
	A2C	A3C	TD3	RD+SAGM	A2C	A3C	TD3	RD+SAGM	A2C	A3C	TD3	RD+SAGM
Clear night	69.34	80.77	81.69	**94.52**	3.28	2.50	2.45	**1.28**	4.16	3.70	3.68	**3.32**
Clear day	72.58	84.16	84.38	**96.87**	2.44	1.96	1.78	**0.96**	2.49	2.08	1.91	**1.76**
Average	70.96	82.47	83.04	**95.70**	2.86	2.23	2.12	**1.12**	3.33	2.89	3.75	**2.54**

TABLE III
VALIDATION METRICS FOR THE PROPOSED APPROACH AND EXISTING SOTA NAVIGATION APPROACHES. THE GROUND TRUTH IS USED AS THE BASELINE FOR ALL COMPUTING RESULTS.

Models	Town 1		Town 2		Town 3		Town 4		Town 5		Town 6	
	RMSE (m) ↓	Times (s) ↓	RMSE (m) ↓	Times (s) ↓	RMSE (m) ↓	Times (s) ↓	RMSE (m) ↓	Times (s) ↓	RMSE (m) ↓	Times (s) ↓	RMSE (m) ↓	Times (s) ↓
Manual control [28]	0.79	45.7	0.84	43.1	0.92	63.4	0.79	55.2	0.75	63.0	0.81	59.4
DQN-Waypoints [29]	0.73	38.9	0.72	36.5	0.77	57.8	0.64	40.1	0.67	56.2	0.69	54.5
DQN-CNN [30]	0.68	35.3	0.61	24.7	0.74	52.5	0.59	39.7	0.58	48.8	0.52	52.9
OGM [31]	0.24	25.2	0.22	24.8	0.26	43.7	0.24	**34.2**	0.30	38.4	0.33	44.7
A3C + RD + SAGM	0.23	24.5	0.21	24.1	0.27	43.9	0.25	36.5	0.32	40.3	0.34	42.7
TD3 + RD + SAGM	0.21	22.4	0.19	22.0	0.25	42.3	0.23	34.8	0.30	37.8	0.31	41.2
DDPG + RD	0.29	30.8	0.25	30.3	0.33	49.6	0.25	39.2	0.37	44.8	0.40	44.7
DDPG + SAGM	0.24	26.4	0.23	26.1	0.29	48.4	0.28	39.6	0.37	42.7	0.39	45.2
DDPG + RD + SAGM (our)	**0.18**(↑ 61%,↑ 33%)	**20.3**	**0.16**(↑ 56%,↑ 44%)	**19.9**	**0.24**(↑ 38%,↑ 21%)	**41.6**	**0.22**(↑ 14%,↑ 27%)	**34.2**	**0.28**(↑ 32%,↑ 32%)	**35.6**	**0.28**(↑ 43%,↑ 39%)	**39.8**

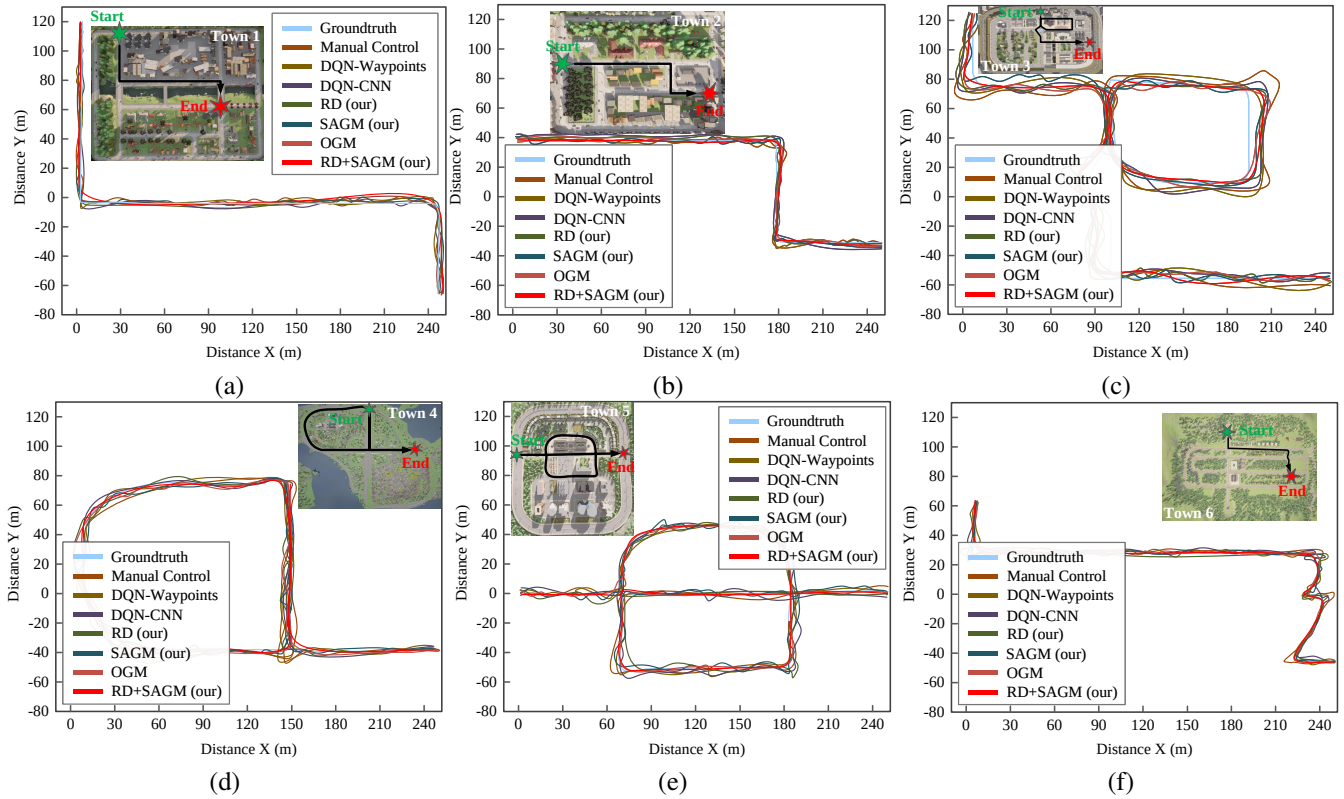

Fig. 11. Multi-scenario tests and comparisons of state-of-the-art (SOTAs) vehicle navigation based on CARLA. (a)-(b) represent simulated scenes, specifically Town 1 to Town 6. The cyan box represents the ground truth of the vehicle navigation without obstacles. Since all dynamic and static obstacles are manually avoided, the path becomes smoother.

the mxnet framework to encapsulate the offline trained model.

We plot the real trajectory of the vehicle and the observed local trajectory of each surrounding static and dynamic obstacle in Fig. 12. Our approach can well understand the motion trend of each dynamic obstacle by learning object-to-vehicle and vehicle-to-object interactions (as shown in the middle example of Fig. 11), and then generate a more reasonable motion strategy to achieve high-efficiency navigation.

We first demonstrate our generalizability to various environments. Then, we evaluate the trained model in unseen scenes with obstacles as shown in Table IV. The results show that: 1) Increased obstacles reduce the navigation success rate; 2) Low illumination reduces the navigation success rate since cameras are susceptible to illumination.

VI. CONCLUSIONS

In this study, we develop a multi-modal deep reinforcement learning (DRL) framework to facilitate smooth and safe navigation for a mobile vehicle in challenging environments. We employ a multi-modal vision fusion strategy to create a locally observable model to address the issue of navigation failure caused by inadequate visual cues in complex environments. More importantly, the model is equipped with RD and SAGM, which enable the intelligent vehicle to make global behavior

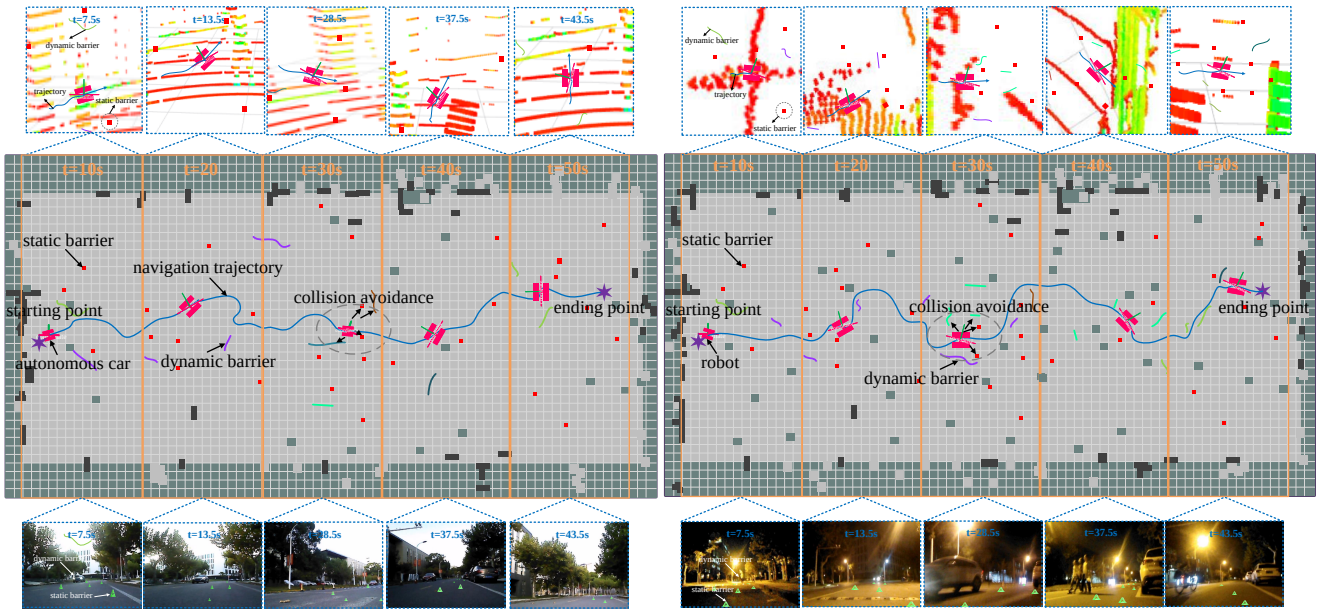

Fig. 12. The intelligent vehicle platform used in our physical experiments for real-world tests. The top of the figure represents the LiDAR scanning thread, the middle represents the real-world environment navigation on the observable ROS platform, and the bottom represents the visual observation thread and current vehicle position. (a) and (b) are the daytime and nighttime navigation tests.

TABLE IV
GENERALIZABILITY OF OUR APPROACH TO A VARIETY OF ENVIRONMENTAL STYLES. THE PARAMETERS IN THE ENVIRONMENT TYPE REPRESENT THE ENVIRONMENT SIZE/STATIC OBSTACLE DENSITY/DYNAMIC OBSTACLE DENSITY. FOR EXAMPLE, $150 \times 4.6/10\%/$RANDOM MEANS THAT THE VEHICLE IS TESTED IN AN ENVIRONMENT WITH A LENGTH AND WIDTH OF 150 METERS AND 4.6 METERS RESPECTIVELY. THERE ARE 10 RANDOMLY PLACED OBSTACLES IN EVERY FIVE-METER PATH IN THE TEST ENVIRONMENT, AND THE DYNAMIC OBSTACLES ARE RANDOM.

Illumination conditions (total 54 tests)	Environmental styles (18 tests for each)	Success rate	Standard Deviation
Daytime scenes	150×4.6/10%/random	**88.9%**	3.076 (±0.011)
	50×4.6/15%/random	77.8%	3.366 (±0.017)
	150×4.6/20%/random	**72.2%**	3.798 (±0.009)
Nighttime scenes	150×4.6/10%/random	83.3%	4.226 (±0.004)
	150×4.6/15%/random	61.1%	4.762 (±0.012)
	50×4.6/20%/random	44.4%	5.83 (±0.003)

decisions based on limited local observations. We conduct comprehensive experiments, including simulations and real-world tests, to evaluate the effectiveness of the proposed navigation method. Additionally, we conduct further ablation studies and component analysis to assess the significance of model parameters and key components of the proposed DRL model.

Potential limitations: Our approach faces challenges in extreme lighting conditions, such as nighttime, and in environments with many dynamic obstacles. Although the multi-modal fusion mechanism improves robustness, visual sensors struggle in low-light, reducing navigation success rates. In high-density obstacle environments, the success rate drops to 44.4%.

Future work: These failure cases reveal opportunities for improvement. First, using night-adapted visual algorithms, like infrared sensing or low-light enhancement, may address low-light limitations. Additionally, precise sensing of obstacle speed and trajectory within the multi-modal framework could enhance navigation in crowded or dynamic environments. Finally, while the cross-domain self-attention mechanism (SAGM) has improved adaptability, challenges remain in transferring capabilities from simulation to real-world scenarios. Integrating more real-world data in future training could further strengthen the model's robustness.

REFERENCES

[1] L. Jin, H. Zhang, C. Ye, "Camera intrinsic parameters estimation by visual–inertial odometry for a mobile phone with application to assisted navigation," IEEE-ASME Transactions on Mechatronics, vol. 25, no. 4, pp. 1803-1811, 2020.

[2] R. Liu, Y. He, C. Yuen, B. P. Lau, R. Ali, W. Fu, Z. Cao, "Cost-effective mapping of mobile robot based on the fusion of UWB and short-range 2-D LiDAR," IEEE-ASME Transactions on Mechatronics, vol. 27, no. 3, pp. 1321-1331, 2021.

[3] Y. Li, Y. Cai, R. Malekian, H. Wang, M A. Sotelo, Z. Li, "Creating navigation map in semi-open scenarios for intelligent vehicle localization using multi-sensor fusion," Expert Systems with Applications, vol. 184, pp. 1-12, 2021.

[4] L. Gao, G. Battistelli, L. Chisci, "PHD-SLAM 2.0: Efficient SLAM in the Presence of Missdetections and Clutter," IEEE Transactions on Robotics, vol. 37, no. 5, pp. 1834-1843, 2021.

[5] C L. Li, K. Sohn, J. Yoon, T. Pfister, "Cutpaste: Self-supervised learning for anomaly detection and localization," In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9664-9674, 2021.

[6] D. Dugas, J. Nieto, R. Siegwart, J J. Chung, "Navrep: Unsupervised representations for reinforcement learning of robot navigation in dynamic human environments," In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 7829-7835, 2021.

[7] L. Chen, X. Hu, B. Tang, Y. Cheng, "Conditional DQN-based motion planning with fuzzy logic for autonomous driving," IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 4, pp. 2966-2977, 2020.

[8] X. Liu, Y. Liu, Y. Chen, L. Hanzo, "Enhancing the fuel-economy of V2I-assisted autonomous driving: A reinforcement learning approach," IEEE Transactions on Vehicular Technology, vol. 69, no. 8, pp. 8329-8342, 2020.

[9] C. Qiu, Y. Hu, Y. Chen, B. Zeng, "Deep deterministic policy gradient (DDPG)-based energy harvesting wireless communications," IEEE Internet of Things Journal, vol. 6, no. 5, pp. 8577-8588, 2019.

[10] S. Li, Y. Wu, X. Cui, H. Dong, F. Fang, S. Russell, "Robust multi-agent reinforcement learning via minimax deep deterministic policy gradient," In Proceedings of the AAAI Conference on Artificial Intelligence, pp. 4213-4220, 2019.
[11] C. Huang, R. Zhang, M. Ouyang, P. Wei, J. Lin, J. Su, L. Lin, "Deductive reinforcement learning for visual autonomous urban driving navigation" IEEE Transactions on Neural Networks and Learning Systems, vol. 32, no. 12, pp. 5379-5391, 2021.
[12] L. Tai, G. Paolo, M. Liu, "Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation," In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 31-36, 2017.
[13] H. Wu, B. Tao, Z. Gong, Z. Yin, H. Ding, "A Standalone RFID-Based Mobile Robot Navigation Method Using Single Passive Tag," IEEE Transactions on Automation Science and Engineering, vol. 18, no. 4, pp. 1529-1537, 2021.
[14] X. Huang, H. Deng, W. Zhang, R. Song, Y. Li, "Towards Multi-Modal Perception-Based Navigation: A Deep Reinforcement Learning Method," IEEE Robotics and Automation Letters, vol. 6, no. 3, pp. 4986-4993, 2021.
[15] J. H. Liu and C. C. Wang, "Vision-Based Obstacle Avoidance for Mobile Robots Using Convolutional Neural Networks," IEEE Transactions on Industrial Electronics, vol. 67, no. 11, pp. 10034-10044, 2019.
[16] J. Wang, Y. Zhang, Y. Zhang, Z. Wang, "Vision-Based Obstacle Avoidance for Mobile Robots Using Reinforcement Learning," IEEE International Conference on Systems, Man and Cybernetics (SMC), pp. 2404-2409, 2019.
[17] J. Liu., "Mobile Robot Obstacle Avoidance Based on Multi-Sensor Fusion and Deep Reinforcement Learning," IEEE Transactions on Industrial Informatics, vol. 16, no. 11, pp. 7170-7180, 2020.
[18] Y. Wang, J. Liu, Y. Liu, Z. Wang, J. Zhang, "A Navigation Framework Based on Multimodal Fusion and Social Awareness for Assistive Robots in Human Inhabited Environments," IEEE Transactions on Cognitive and Developmental Systems, vol. 12, no. (4), pp. 1037-1051, 2021.
[19] Y. Xiao, J. Zhang, Z. Wang, Y. Liu, "Vision-Based Mobile Robotics Obstacle Avoidance with Deep Reinforcement Learning," IEEE International Conference on Robotics and Automation (ICRA), pp. 11288-11294, 2021.
[20] X. Yu, B. Zhou, Z. Chang, K. Qian, F. Fang, "MMDF: Multi-Modal Deep Feature Based Place Recognition of Mobile Robots with Applications on Cross-Scene Navigation," IEEE Robotics and Automation Letters, vol. 7, no. 3, pp. 6742-6749, 2022.
[21] N. Ü. Akmandor, H. Li, G. Lvov, E. Dusel, T. Padir, "Deep Reinforcement Learning based Robot Navigation in Dynamic Environments using Occupancy Values of Motion Primitives," In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 11687-11694, 2022.
[22] D. Quillen, E. Jang, O. Nachum, C. Finn, J. Ibarz, S. Levine, "Deep Reinforcement Learning for Vision-Based Robotic Grasping: A Simulated Comparative Evaluation of Off-Policy Methods," In Proceedings of the International Conference on Robotics and Automation (ICRA), pp. 6284-6291, 2018.
[23] W. Zhu, M. Hayashibe, "A Hierarchical Deep Reinforcement Learning Framework with High Efficiency and Generalization for Fast and Safe Navigation," IEEE Transactions on Industrial Electronics, vol. 70, no. 5, pp. 4962-4971, 2023.
[24] Y. H. Khalil, H. T. Mouftah, "Exploiting Multi-Modal Fusion for Urban Autonomous Driving Using Latent Deep Reinforcement Learning," IEEE Transactions on Vehicular Technology, 2023, vol. 72, no. 3, pp. 2921-2935.
[25] Z. Li, A. Zhou. "RDDRL: A Recurrent Deduction Deep Reinforcement Learning Model for Multimodal Vision-robot Navigation". Applied Intelligence, 2023, vol. 53, no. 20, pp. 23244-23270.
[26] X. Wang, Q. Huang, A. Celikyilmaz, J. Gao, D. Shen, Y. Wang, L. Zhang. "Reinforced Cross-modal Matching and Self-supervised Imitation Learning for Vision-Language Navigation". In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 6629-6638.
[27] X. Zhao, C. Weber, M. B. Hafez, S. Wermter, "Impact Makes a Sound and Sound Makes an Impact: Sound Guides Representations and Explorations," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Kyoto, Japan, 2022, pp. 2512-2518
[28] K. Wu, H. Wang, M. Abolfazli Esfahani, S. Yuan, "BND*-DDQN: Learn to Steer Autonomously Through Deep Reinforcement Learning," IEEE Transactions on Cognitive and Developmental Systems, 2021, vol. 13, no. 2, pp. 249-261.
[29] M. Burhan Hafez and S. Wermter, "Behavior Self-Organization Supports Task Inference for Continual Robot Learning," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Prague, Czech Republic, 2021, pp. 6739-6746.
[30] Y. Zou, P. Ji, Q H. Tran, J B. Huang, M. Chandraker, "Learning monocular visual odometry via self-supervised long-term modeling," In Proceedings of the European Conference on Computer Vision (ECCV), pp. 710-727, 2020.
[31] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, P., Hazirbas, V. Golkov, T. Brox, "Flownet: Learning optical flow with convolutional networks," In Proceedings of the IEEE International Conference on Computer Vision, pp. 2758-2766, 2015.
[32] A. Paszke, A. Chaurasia, S. Kim, E. Culurciello, "Enet: A deep neural network architecture for real-time semantic segmentation," arXiv preprint arXiv:1606.02147, pp. 1-10, 2016.
[33] O. Chum and J. Matas, "Matching with PROSAC - progressive sample consensus," IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Diego, USA, 2005, pp. 220-226.
[34] D. Wisth, M. Camurri, S. Das, M. Fallon, "Unified Multi-Modal Landmark Tracking for Tightly Coupled Lidar-Visual-Inertial Odometry," IEEE Robotics and Automation Letters, vol. 6, no. 2, pp. 1004-1011, 2021.
[35] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, V. Koltun, "CARLA: An open urban driving simulator," In Proceedings of the Conference on robot learning, pp. 1-16, 2017.
[36] Ó. Pérez-Gil, R. Barea, E. López-Guillén, L. M. Bergasa, P. A. Revenga, R. Gutiérrez, A. Díaz, "DQN-based deep reinforcement learning for autonomous driving," In Workshop of Physical Agents, 2020, pp. 60–76.
[37] Ó. Pérez-Gil, R. Barea, E. López-Guillén, L. M. Bergasa, C. Gomez-Huelamo, R. Gutiérrez, A. Diaz-Diaz, "Deep reinforcement learning based control for Autonomous Vehicles in CARLA. Multimedia Tools and Applications," 2022, vol. 81, no. 3, pp. 3553-76.
[38] M. Eraqi, M. N. Moustafa and J. Honer, "Dynamic Conditional Imitation Learning for Autonomous Driving," in IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 12, pp. 22988-23001, Dec. 2022, pp. 1-14.

Zhenyu Li (IEEE Member) received the Ph.D. degree in Mechanical Engineering from Tongji University, Shanghai, China, in 2023. He is currently a lecturer with the Qilu University of Technology and an Assistant Professor with the Shandong Academy of Sciences. His current research interests include intelligent perception, visual localization, and navigation for robot automation in complex environments. He won the "Best Paper Finalist" in the 2019 IEEE-ROBIO Conference Selection. He has published over 30 papers including IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, IEEE TRANSACTIONS ON ARTIFICIAL INTELLIGENCE, etc., and was the reviewer for several IEEE TRANSACTIONS JOURNALS, such as IEEE TII, IEEE TMECH, IEEE TGRS, etc.

Tianyi Shang is currently a sophomore at Fuzhou University in China, pursuing a Bachelor's degree in Electronic Information Engineering. Among the cohort of 91 students in the class of 2024, Tianyi has achieved a cumulative GPA of 3.85, ranking third, showcasing his exceptional academic talents. His research efforts are primarily focused on the field of computer vision. Tianyi's interests include intelligent perception, visual localization, and navigation, all of which are crucial for enhancing the autonomy and efficiency of robots. In addition, he is exploring metaheuristic algorithms, which are essential for solving optimization problems within his research domain. Under the guidance of Zhenyu Li, So far, as a visiting student of CV4RA lab., Tianyi has submitted two outstanding works to CVPR-2025 and IEEE ROBOTICS AND AUTOMATION LETTERS, and participated in a paper published in IEEE-TII.

Pengjie Xu received the B.S., M.S., and Ph.D. degrees from Shandong University of Technology, Qingdao University, and Tongji University, China in 2015, 2018, and 2023, respectively. Currently, he is a postdoctoral fellow with the School of Mechanical Engineering, Shanghai Jiao Tong University, China. His research interests include machine learning and robotics systems.

Wenhao Li is currently a sophomore at Qilu University of Technology, pursuing a Bachelor's degree in Intelligent Manufacturing Engineering. Among the cohort of 51 students in the class of 2024, Wenhao has achieved a cumulative GPA of 3.99, ranking 4th. Wenhao's interests include computer vision. Since joining the CV4RA lab. led by Zhenyu Li in October 2022, Wenhao has participated in the research of three academic papers, including the recent CVPR-2025. In addition, under the guidance of Zhenyu Li, Wenhao successfully won the "2024 Undergraduate Innovation and Entrepreneurship Project" and won the first prize in the "2024 Statistical Modeling Competition".