

Partial Attention in Deep Reinforcement Learning for Safe Multi-Agent Control

Turki Bin Mohaya Peter Seiler

Abstract—Attention mechanisms excel at learning sequential patterns by discriminating data based on relevance and importance. This provides state-of-the-art performance in today’s advanced generative artificial intelligence models. This paper applies this concept of an attention mechanism for multi-agent safe control. We specifically consider the design of a neural network to control autonomous vehicles in a highway merging scenario. The environment is modeled as a Decentralized Partially Observable Markov Decision Process (Dec-POMDP). Within a QMIX framework, we include partial attention for each autonomous vehicle, thus allowing each ego vehicle to focus on the most relevant neighboring vehicles. Moreover, we propose a comprehensive reward signal that considers the environment’s global objectives (e.g., safety and vehicle flow) and the individual interests of each agent. Simulations are conducted in the Simulation of Urban Mobility (SUMO). The results show better performance compared to other driving algorithms in terms of safety, driving speed, and reward.

I. INTRODUCTION

Highway merging is a fundamental yet challenging problem in autonomous driving. It requires agents to reason about dynamic interactions under uncertainty, where safe and efficient decisions depend not only on the agent’s own state but also on the behaviors of surrounding vehicles. Traditional rule-based methods, although simple, often lack the flexibility to adapt to complex and dynamic merging situations. On the other hand, fully centralized deep reinforcement learning approaches struggle with scalability and are difficult to deploy in practical settings. Thus, decentralized policies that can selectively attend to relevant information present a promising direction for improving highway merging.

There are several similar works in the literature that deploy Multi-Agent Reinforcement Learning (MARL) [1], [2], [3], [4], [5]. The authors in [6] proposed a simple recurrent unit to capture temporal patterns in the highway merging problem. These patterns are then fed to a Deep Deterministic Policy Gradient (DDPG) [7] network. After that, they enhance the training by introducing a prioritized replay buffer that samples experiences in proportion to the error of performance during that specific experience. This allows frequent replay of challenging scenarios for fast learning. The work in [8] uses cross-attention mechanisms to fuse pose data and semantic data from different instruments on the vehicle, mainly for navigation. Their work shows good performance, but requires costly computational resources due to the complexity of the overall proposed framework.

T. Bin Mohaya and P. Seiler are with the Department of Electrical Engineering and Computer Science at the University of Michigan, Ann Arbor, MI 48109, USA. Email: {turki,pseiler}@umich.edu. The authors acknowledge funding from the Ford Motor Company.

We consider the specific multi-agent scenario where vehicles are required to merge onto a highway. A model for this highway merging task is described in Section II. Our proposed solution for autonomous merging builds on two ingredients that are reviewed in Section III: attention mechanisms and QMIX [9] for decentralized MARL. Section IV then details our partial attention design and reward shaping. We illustrate our proposed method using the Simulation of Urban Mobility (SUMO) [10] and compare against other approaches (Section V). Finally, Section VI concludes the paper and discusses potential future directions.

Our contributions are twofold. First, we enhance the QMIX architecture by introducing a partial attention mechanism that focuses on the most critical interactions for each agent. Our notion of partial attention constitutes two elements: spatial attention and temporal attention. In spatial attention, we impose, by design, that each agent only observes the vehicle in front and the vehicle on the opposite merging road. In temporal attention, the neural network learns by itself to automatically focus on past time steps of these vehicles. This improves decision quality without incurring significant computational overhead. Second, we design a comprehensive reward structure that balances individual objectives, such as velocity maintenance and comfort, with global objectives such as collision avoidance and traffic flow improvement. Our method demonstrates improved safety and efficiency, validated through sophisticated simulations.

II. PROBLEM FORMULATION

A. Description

The highway merging problem is challenging as agents with lower velocities approach other agents that drive with fast velocity profiles. Furthermore, it is safety-critical and typically requires a decentralized solution. Fig. 1 (left) illustrates the problem: vehicle B is merging while other highway vehicles are driving with higher velocities. The individual objectives of each agent mainly include safety (i.e., avoiding collision), and maintaining a desired velocity without altering the riders’ comfort or compromising fuel efficiency. However, there are global objectives for the traffic that are sometimes in conflict with the objectives of individual vehicles. For example, it is desired to maintain a high average velocity to enhance vehicle flow, but highway agents may need to decelerate to allow merging. Also, the merging road should increase its throughput without negatively affecting the highway’s average velocity.

Each agent makes decisions, in principle, based on all other nearby vehicles. However, not all other vehicles are

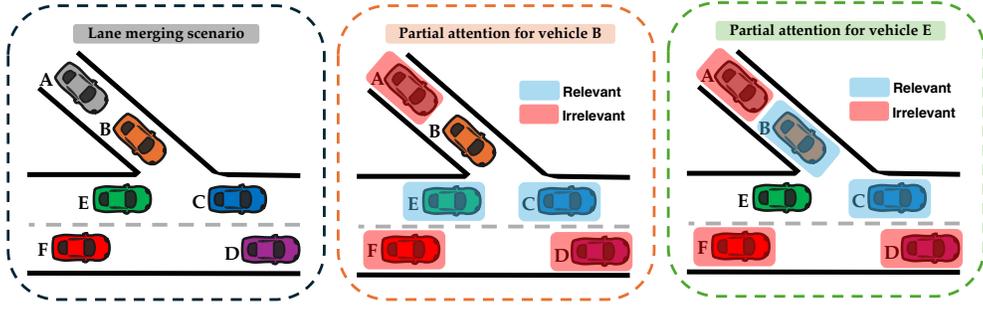


Fig. 1: **Left:** The highway merging problem. **Middle and Right:** Our contribution is deploying partial attention to the most critical interactions for safe highway merging.

equally relevant to an agent’s decision. For example, in Fig. 1 (middle), vehicles A, F, and D are less relevant to the decision of vehicle B. On the other hand, vehicles C and E are of great importance to the merger decision. In fact, considering only information from these two agents can significantly reduce the computational cost for the merging decision-making process, and thus simplify the multi-agent highway merging problem. The dual is also true. In Fig. 1 (right), Vehicle E should focus on vehicles B and C while reducing attention to other vehicles.

B. POMDPs

The multi-vehicle environment is modeled as a Decentralized Partially Observable Markov Decision Process (Dec-POMDP) [11] and represented by the following tuple:

$$\mathcal{G} = (\mathcal{I}, \mathcal{S}, \{\mathcal{A}_i\}, \mathcal{T}, r, \{\Omega_i\}, \mathcal{O}, \gamma). \quad (1)$$

The index set $\mathcal{I} := \{1, \dots, N\}$ labels the N vehicles. A state $s \in \mathcal{S}$ summarizes the status of the highway. Each vehicle $i \in \mathcal{I}$ selects a discrete control $a_i \in \mathcal{A}_i$. Collecting the local controls yields the joint control $a = (a_1, \dots, a_N) \in \mathcal{A}$, with $\mathcal{A} = \times_{i \in \mathcal{I}} \mathcal{A}_i$, where \times denotes the Cartesian product. State evolution is governed by the probability kernel $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$, so that $\mathcal{T}(s, a, s') = \Pr(s' | s, a)$, where s' denotes the next state obtained after applying the action a at state s . After the transition, vehicle i receives a private observation $o_i \in \Omega_i$. The stacked observation $o = (o_1, \dots, o_N)$ resides in $\Omega = \times_{i \in \mathcal{I}} \Omega_i$. A measurement function $\mathcal{O} : \mathcal{S} \times \mathcal{A} \times \Omega \rightarrow [0, 1]$ defines the likelihood of an observation o for a given state-action pair (s, a) : $\mathcal{O}(s, a, o) = \Pr(o | s, a)$. Lastly, a scalar reward signal $r : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$ provides the reward $r(s, a, s')$ accrued on a transition from (s, a) to s' . Future scalar rewards are weighted by the discount factor $\gamma \in [0, 1]$. Specifically, let G_t denote the total discounted reward starting at time t and going through the finite horizon T of an episode. This total discounted reward is given by:

$$G_t = \sum_{k=t}^{T-1} \gamma^{k-t} r(k+1). \quad (2)$$

Here, $r(k+1)$ is a shortened notation for the reward accrued at time $k+1$ for the transition from $(s(k), a(k))$ to the next state $s(k+1)$. The finite-horizon return can be equivalently expressed in recursive form as: $G_t = r(t+1) +$

γG_{t+1} , $G_T = 0$, where $r(t+1)$ is the immediate reward and γG_{t+1} is the discounted future reward. The terminal condition $G_T = 0$ enforces the final horizon.

III. BACKGROUND

Deep neural networks [12] have shown an astonishing performance in learning complex patterns across vast dynamics and datasets. This is due to their ability to automatically extract features and optimize their weights through backpropagating the error loss. However, automatic feature extraction can, in some problems, suffer from an inability to focus on more relevant parts of the data. This can lead to sub-optimal solutions and inefficient training. Therefore, attention mechanisms were designed to equip neural networks with the ability to learn selective focus. This enables state-of-the-art performance in language translation and large language models [13], [14]. We propose to apply similar attention mechanisms to autonomous driving.

A. Attention Neural Networks

Consider a vector-valued sequence $\{s_0, s_1, \dots, s_T\} \subset \mathbb{R}^n$, and stack it row-wise to form the matrix $S := [s_0^\top; s_1^\top; \dots; s_T^\top] \in \mathbb{R}^{(T+1) \times n}$. The scaled attention neural network [13] takes S as input and produces an output matrix $Z \in \mathbb{R}^{(T+1) \times d_o}$ where the dimension d_o is described below. The output is constructed from H attention heads, each computing a query, key, and value. For head $j = 1, \dots, H$, the query $\mathcal{Q}_j \in \mathbb{R}^{(T+1) \times d_j}$, key $\mathcal{K}_j \in \mathbb{R}^{(T+1) \times d_j}$, and value $\mathcal{V}_j \in \mathbb{R}^{(T+1) \times d_j}$ are computed as

$$\mathcal{Q}_j = SW_Q^j, \quad \mathcal{K}_j = SW_K^j, \quad \mathcal{V}_j = SW_V^j, \quad (3)$$

where $W_Q^j, W_K^j, W_V^j \in \mathbb{R}^{n \times d_j}$ are the learned projection weights for the j -th head. Setting $H = 1$ yields the single-head attention neural network, while setting $H > 1$ employs multiple attention heads in parallel. Each head independently processes the input sequence, allowing the model to capture different aspects of the input data simultaneously. The attention weights for each head are $\mathcal{A}_j = \text{Softmax}\left(\frac{\mathcal{Q}_j \mathcal{K}_j^\top}{\sqrt{d_j}}\right) \in \mathbb{R}^{(T+1) \times (T+1)}$, where d_j is the dimension of the keys and queries. The output of head j is then $Z_j = \mathcal{A}_j \mathcal{V}_j \in \mathbb{R}^{(T+1) \times d_j}$. The outputs of all heads are concatenated and linearly transformed to produce the final output matrix of the attention neural network:

$$Z = [Z_1 \quad Z_2 \quad \dots \quad Z_H] W_O \in \mathbb{R}^{(T+1) \times d_o}, \quad (4)$$

where $d = \sum_{j=1}^H d_j$ is the column dimension of the concatenated matrix. Moreover, $W_O \in \mathbb{R}^{d \times d_o}$ is a learned projection matrix that maps the combined output to the desired output dimension d_o . This multi-head approach enables the model to capture diverse patterns and relationships within the input sequence. This enhances its ability to learn complex dependencies and improve overall performance.

B. QMIX

Multi-agent deep reinforcement learning (MADRL) provides a robust method to learn practical policies or controllers for stochastic multi-agent environments. This is due to their ability to capture highly non-linear stochastic dynamics by iteratively interacting with the environment and observing the consequences. Independent Q-learning (IQL) [15] decomposes the multi-agent problem into parallel single-agent problems that are simultaneously collocated. In IQL, each agent assumes the other agents are non-moving in the environment. However, learning for the agents must be coordinated to address the nonstationary environment.

The principle of optimality [16] can be used to express a recursive (centralized) solution for the total action-value function Q_{tot} of all agents. QMIX [9] approximates the total action-value function through a nonlinear mapping $\text{Mix}(\cdot)$ of their individual action-value functions $\{Q_i\}_{i=1}^N$ of the N agents. This nonlinear mapping is implemented as a neural network and is given by

$$Q_{tot}(o, a; \theta) = \text{Mix}(o, a, Q_1(o_1, a_1; \theta_1), \dots, Q_N(o_N, a_N; \theta_N), \theta_{N+1}), \quad (5)$$

where θ includes the weights of the individual action-value functions $\{\theta_i\}_{i=1}^N$ and the weights of the mixing network θ_{N+1} .

QMIX minimizes a Deep Q-Network [17] regression loss over a replay mini-batch of size B . To define this loss, let $o(m)$ and $a(m)$ denote the joint observation and the joint action at sample m , respectively. The one-step target is

$$y(m) = r(m) + \gamma \max_{a'} Q_{tot}(o'(m), a'; \theta^-), \quad (6)$$

where $r(m)$ is the immediate reward, $o'(m)$ is the next joint observation at sample m , a' is the next joint action, and θ^- are the parameters of a slowly updated target network. The loss over a replay mini-batch of size B is then given by

$$\mathcal{L}(\theta) = \frac{1}{B} \sum_{m=1}^B [y(m) - Q_{tot}(o(m), a(m); \theta)]^2. \quad (7)$$

The mixing network weights are constrained to be non-negative. This architectural choice enforces the monotonicity condition $\frac{\partial Q_{tot}(o, a)}{\partial Q_i(o_i, a_i)} \geq 0, \forall i \in \mathcal{I}$, ensuring that an improvement in any agent's value cannot reduce the global estimate. As a result of Q_{tot} being monotone, the joint maximizer factorizes into independent maximizers. This is due to the Individual-Global-Max (IGM) principle [18]:

$$\arg \max_a Q_{tot}(o, a) = \begin{bmatrix} \arg \max_{a_1} Q_1(o_1, a_1) \\ \vdots \\ \arg \max_{a_N} Q_N(o_N, a_N) \end{bmatrix}, \quad (8)$$

so team-optimal actions can be obtained by greedy choices made locally by each agent.

During learning, the full joint trajectory o is available to the mixing network, providing a centralized viewpoint that removes the non-stationarity that is a consequence of purely decentralized training. In this phase, each agent selects a random action a_i^{random} with probability ϵ to encourage exploration, while with probability $1 - \epsilon$ it selects the greedy action $a_i^* = \arg \max_{a_i} Q_i(o_i, a_i)$. Formally, the individual action selection at each step is

$$a_i = \begin{cases} a_i^{\text{random}} & \text{with probability } \epsilon, \\ a_i^* & \text{with probability } 1 - \epsilon. \end{cases} \quad (9)$$

The exploration probability ϵ decays over time according to a fixed decay rate ϵ_{decay} until reaching a specified minimum value ϵ_{min} . During evaluation, the model fully exploits its learned policy by setting $\epsilon = 0$, thereby always choosing the greedy action. After convergence, the mixing network is discarded, and each agent selects its action as a_i^* using local information o_i . It is worth noting that even with each agent selecting actions based on local information, (8) guarantees that the resulting joint action remains optimal.

IV. APPROACH

A. Agent State Design

The state of each autonomous vehicle (AV) is designed to encapsulate both its own dynamics and the dynamics of its immediate surroundings through the partial attention mechanism. This approach allows the i -th AV to focus on the most relevant interactions, specifically the vehicle directly ahead of it, $f(i)$, and the vehicle approaching from the opposite road, $o(i)$. To illustrate, vehicle E in Fig. 1 (right) has a front vehicle C and an opposite vehicle B. On the other hand, in Fig. 1 (middle), vehicle B has the front vehicle C and the opposite vehicle E. Similarly, vehicle A has the front vehicle B and the opposite vehicle E. Finally, vehicles D and C have neither front nor opposite vehicle. Every agent's state representation is composed of its current state and the historical states of these two neighboring vehicles over a specified time window. The *kinematic state* of vehicle i at time t is defined as $\bar{S}_i(t) = [x_i(t) \ y_i(t) \ v_i(t) \ a_i(t)]^\top \in \mathbb{R}^4$, where $x_i(t)$ and $y_i(t)$ are the position coordinates, $v_i(t)$ is the speed, and $a_i(t)$ is the acceleration of the vehicle.

The *information state* of vehicle i at time t , denoted $S_i(t) \in \mathbb{R}^{4+8(w+1)}$, is defined as

$$S_i(t) = [\bar{S}_i(t)^\top \ \text{vec}(F_i(t))^\top \ \text{vec}(O_i(t))^\top]^\top, \quad (10)$$

where $\text{vec}(\cdot)$ denotes row-major vectorization of a matrix into a flat vector, and $F_i(t), O_i(t) \in \mathbb{R}^{(w+1) \times 4}$ represent the historical kinematic states of the front and opposite vehicles over the most recent $w + 1$ time steps:

$$F_i(t) = [\bar{S}_{f(i)}(t-w) \ \dots \ \bar{S}_{f(i)}(t)]^\top, \quad (11)$$

$$O_i(t) = [\bar{S}_{o(i)}(t-w) \ \dots \ \bar{S}_{o(i)}(t)]^\top. \quad (12)$$

Here, $\bar{S}_{f(i)}(t)$ and $\bar{S}_{o(i)}(t)$ denote the kinematic states of the front and opposite vehicles associated with agent i ,

respectively. We assume that each agent can perfectly observe the kinematic state of its front and opposite vehicles. This corresponds to an idealized vehicle-to-vehicle (V2V) communication or sensing model.

B. Partial Attention

The front and opposite vehicle histories, $F_i(t)$ and $O_i(t)$, are leveraged by processing through a multi-head attention mechanism. This processing captures temporal dependencies and contextual interactions. This constructs a comprehensive state representation for each agent. The design begins with layer normalization [19]. Given a vector x , the output y of the layer normalization operation can be expressed as

$$y = \frac{x - E[x]}{\sqrt{\text{Var}[x] + \rho}} \odot \kappa + \beta, \quad (13)$$

where $E[x]$ is the mean of x , $\text{Var}[x]$ is the variance of x , and \odot represents element-wise multiplication. We set $\rho = 10^{-5}$ while κ and β are affine learnable parameters. We utilize PyTorch [20] to conduct this operation. The deployment of this function is discussed next.

Each historical sequence is first normalized to ensure consistent scaling across different features and time steps:

$$\tilde{F}_i(t) = \text{LN}(F_i(t)), \quad \tilde{O}_i(t) = \text{LN}(O_i(t)), \quad (14)$$

where $\text{LN}(\cdot)$ denotes layer normalization applied to each feature vector within the sequence. Following normalization, each token in the sequence is projected into a higher-dimensional embedding space to enhance the capacity of the attention mechanism:

$$X_{F_i}(t) = \tilde{F}_i(t) W_F^\top + \mathbf{1}_{w+1} b_F^\top \in \mathbb{R}^{(w+1) \times d_{\text{model}}}, \quad (15)$$

$$X_{O_i}(t) = \tilde{O}_i(t) W_O^\top + \mathbf{1}_{w+1} b_O^\top \in \mathbb{R}^{(w+1) \times d_{\text{model}}}, \quad (16)$$

where $W_F, W_O \in \mathbb{R}^{d_{\text{model}} \times 4}$ and $b_F, b_O \in \mathbb{R}^{d_{\text{model}}}$ denote the weight matrices and bias vectors of the corresponding linear transformations, and $\mathbf{1}_{w+1} \in \mathbb{R}^{w+1}$ is a vector whose entries are all ones. Here, d_{model} is the dimensionality of the embedding space. Next, the projected sequences $X_{F_i}(t)$ and $X_{O_i}(t)$ serve as inputs to separate multi-head attention modules tailored for the front and opposite vehicle histories:

$$Z_{F_i}(t) = \text{MHA}(X_{F_i}(t)) \in \mathbb{R}^{(w+1) \times d_{\text{model}}}, \quad (17)$$

$$Z_{O_i}(t) = \text{MHA}(X_{O_i}(t)) \in \mathbb{R}^{(w+1) \times d_{\text{model}}}. \quad (18)$$

The function $\text{MHA}(\cdot)$ denotes the multi-head attention operation on the input sequence. This mechanism allows the model to focus on different parts of the input sequences simultaneously, capturing diverse aspects of temporal dependencies.

To synthesize the information captured by the attention mechanism, a weighted aggregation of the attention outputs is performed along the temporal axis. This aggregation gives more emphasis on the most recent historical data, reflecting their higher relevance in the current decision-making context. We define $\alpha = \text{Softmax}(\Psi_{0.5,1}) \in \mathbb{R}^{w+1}$ where $\text{Softmax}(\cdot)$ normalizes the vector $\Psi_{0.5,1}$, which is

an increasing evenly spaced vector between 0.5 and 1.0 over the $w + 1$ time steps. Then, the weighted embeddings are

$$E_{F_i}(t) = Z_{F_i}(t)^\top \alpha, \quad E_{O_i}(t) = Z_{O_i}(t)^\top \alpha. \quad (19)$$

This contracts the temporal axis of $Z_{F_i}(t)$ and $Z_{O_i}(t)$, producing a single embedding vector that weights the relevance of each time step. To further enhance the expressive power of the embeddings $E_{F_i}(t)$ and $E_{O_i}(t)$, they are passed through two feed-forward neural networks (FFN) with residual connections and layer normalization as

$$E'_{F_i}(t) = \text{LN}(E_{F_i}(t) + \text{FFN}_F(E_{F_i}(t))) \in \mathbb{R}^{d_{\text{model}}}, \quad (20)$$

$$E'_{O_i}(t) = \text{LN}(E_{O_i}(t) + \text{FFN}_O(E_{O_i}(t))) \in \mathbb{R}^{d_{\text{model}}}, \quad (21)$$

where $\text{FFN}_F(\cdot)$ and $\text{FFN}_O(\cdot)$ are feed-forward networks comprising linear transformations and non-linear activations. The residual connections facilitate gradient flow and stabilize training by allowing the model to retain information from earlier layers.

The final state representation $S'_i(t)$ for agent i at time step t is constructed by concatenating the agent's own state vector $\bar{S}_i(t)$ with the aggregated embeddings from the front and opposite vehicle histories:

$$S'_i(t) = [\bar{S}_i(t)^\top \quad E'_{F_i}(t)^\top \quad E'_{O_i}(t)^\top]^\top \in \mathbb{R}^{4+2d_{\text{model}}}. \quad (22)$$

This enriched state vector integrates both the intrinsic dynamics of the agent and the contextual information derived from its immediate vehicular environment through partial attention. The comprehensive state $S'_i(t)$ is subsequently fed into the QMIX utility network, which leverages this information to estimate the utility values for each possible action, thereby facilitating coordinated and efficient decision-making during merging maneuvers.

C. Reward Design

The reward structure is designed to incentivize desirable behaviors and penalize undesirable actions. It integrates both global and local reward components, ensuring that individual agent performance aligns with overall traffic efficiency and safety objectives. The sum of all terms gives a comprehensive reward signal that facilitates effective learning of the QMIX model and enables coordinated decision-making in the merging scenario.

Global Reward Signal: The global reward is designed to promote collective traffic efficiency and safety through metrics that reflect the overall performance of all agents. First, to promote safety, a penalty is imposed when a collision occurs via the term

$$r_{\text{collision}}(t) = -c_1 N_{\text{collision}}(t), \quad (23)$$

where $N_{\text{collision}}(t)$ represents the number of collisions at time step t , and c_1 is a positive weighting coefficient. Upon receiving this penalty, the episode is terminated. Next, to maintain efficient traffic flow and minimize congestion, a traffic flow term encourages agents to sustain desired speeds:

$$r_{\text{flow}}(t) = c_2 \bar{v}_{\text{highway}}(t) + c_3 \bar{v}_{\text{merging}}(t), \quad (24)$$

where $\bar{v}_{\text{highway}}(t)$ and $\bar{v}_{\text{merging}}(t)$ denote the average velocities on the highway and merging lanes at time step t , and c_2, c_3 are positive balancing coefficients. To mitigate idling and reduce travel time, we define the waiting penalty as

$$r_{\text{waiting}}(t) = -c_4 T_{\text{waiting}}(t), \quad (25)$$

where $T_{\text{waiting}}(t)$ is the cumulative time vehicles spend below the minimum allowed velocity, and c_4 is the weighting coefficient. Finally, to encourage route completion, we define

$$r_{\text{goal}}(t) = c_5 N_{\text{goal}}(t), \quad (26)$$

where $N_{\text{goal}}(t)$ denotes the number of vehicles that successfully reached their destinations at time step t , and c_5 is the corresponding weighting coefficient.

Individual Reward Signal: In addition to the global reward, individual rewards are directly attributed to each agent based on its specific actions and states, ensuring independent performance maximization. Each agent is rewarded for tracking its desired velocity via

$$r_{\text{velocity},i}(t) = -c_6 \frac{|v_i(t) - v_{i,\text{desired}}|}{v_{i,\text{desired}}}, \quad (27)$$

where $v_i(t)$ is the velocity of agent i at time t , $v_{i,\text{desired}}$ is its target velocity, and c_6 is a positive weighting coefficient. Fuel-efficient driving is promoted through

$$r_{\text{efficiency},i}(t) = -c_7 \text{Fuel}_i(t), \quad (28)$$

where $\text{Fuel}_i(t)$ is the fuel consumption of agent i at time step t , and c_7 is the weighting coefficient. Finally, passenger comfort is enforced by penalizing sharp accelerations via

$$r_{\text{comfort},i}(t) = -c_8 |a_i(t)|, \quad (29)$$

where $a_i(t)$ is the acceleration of agent i at time step t , and c_8 is the corresponding weighting coefficient. *Final Reward Signal:* The comprehensive reward $r(t)$ at step t is the sum of all global and individual agent reward terms:

$$r(t) = r_{\text{collision}}(t) + r_{\text{flow}}(t) + r_{\text{waiting}}(t) + r_{\text{goal}}(t) + \sum_{i=1}^N r_{\text{velocity},i}(t) + r_{\text{efficiency},i}(t) + r_{\text{comfort},i}(t). \quad (30)$$

The total reward $r(t)$ enters the discounted return defined in (2). This reward structure ensures that agents are simultaneously motivated to maintain safe and efficient driving, both individually and collectively, enhancing overall traffic dynamics during highway merging.

V. RESULTS

We utilize the Simulation of Urban Mobility (SUMO) [10] to build the highway environment. Neural networks are designed and trained via PyTorch [20] and then deployed in SUMO through the integration of TraCI [21]. Our model is trained for 1000 episodes, each with a maximum of 1000 time steps. The hyperparameters used to train the networks are summarized in Table I.

The highway has two lanes and a length of 400 meters, but vehicles are not allowed to change lanes. The merging

TABLE I: Training Hyperparameters

Parameter	Value
Episodes	1000
Maximum time steps per episode	1000
Optimizer	AdamW [22]
B (batch size)	256
γ (discount factor)	0.99
Learning rate	0.0001
ϵ	1.0
ϵ_{min}	0.05
ϵ_{decay}	0.99
Target network update interval	4 episodes
TraCI sampling interval (s)	0.1
Replay buffer size	1000000
Action space - acceleration (m/s ²)	$[-6, -3, -2, -1, 0, 1, 2, 3, 6]$
w	9

TABLE II: Vehicle Generation Parameters

Parameter	Value
Number of agents	16
Initial velocity for highway agents (m/s)	$\sim \text{Uniform}([7, 10])$
Initial velocity merging agents (m/s)	$\sim \text{Uniform}([4, 8])$
Departure time (s)	$\sim \text{Uniform}([0, 100])$
Route	$\sim \text{Uniform}(\{HW, M\})$
Vehicle length (m)	5.0

road has a length of 100 meters and one lane. Table II reports the random parameters used to generate vehicles for each episode. This includes a fixed number of agents, the initial velocities of highway and merging road vehicles, the assigned departure time (the time at which the vehicle begins to exist), and the assigned road, i.e., label HW for highway spawn and M for a merging road spawn.

Table III reports the reward coefficients. c_1 is large to critically prioritize vehicle safety. c_2 and c_3 are balanced to enhance both the merging flow and the highway flow. c_4 penalizes waiting time without overshadowing other terms. c_5 rewards safely reaching the goal, but it is not enough to receive a large reward. c_6 increases the vehicle's velocity, while c_7 is designed not to exceed high speeds. Lastly, the comfort term c_8 enables smoother acceleration signals.

The training was conducted on a MacBook Pro equipped with an Apple M4 chip, and it lasted approximately 56 minutes. Fig. 2 shows the performance of the proposed method during the training phase. Within 500 episodes, the model converged to high average velocities while significantly reducing the average number of collisions. Fuel consumption increases during training as a result of increasing the average velocities of vehicles. Furthermore, we conduct an ablation study by training the model without the temporal attention layers (i.e, omitting (17) and (18)). This ablation comparison is referred to as Vanilla QMIX (VQMIX) in Fig. 2. Its performance diverges within 300 episodes of training. This illustrates the importance of the temporal attention layers to focus on the critical temporal dynamics. Hence, the ablation study demonstrates the effectiveness of the proposed method.

Next, we discuss the results during the evaluation phase. Fig. 3 compares our Partial Attention QMIX model with SUMO's Intelligent Driving Model (IDM) [23] in four

TABLE III: Reward Coefficients

Coefficient	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8
Value	40	0.5	0.9	1.0	1.0	3.0	0.00001	0.01

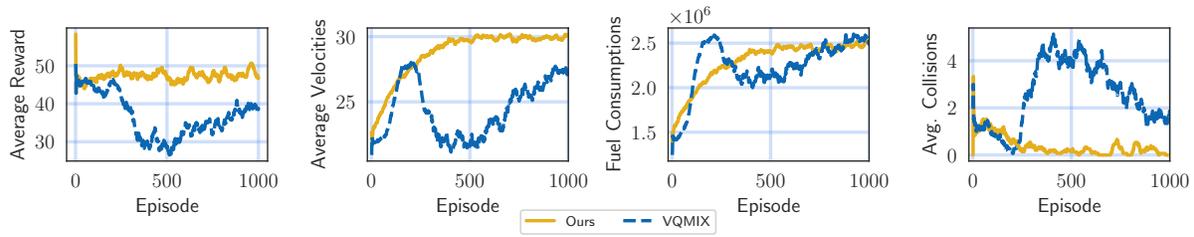


Fig. 2: Performance during the training phase.

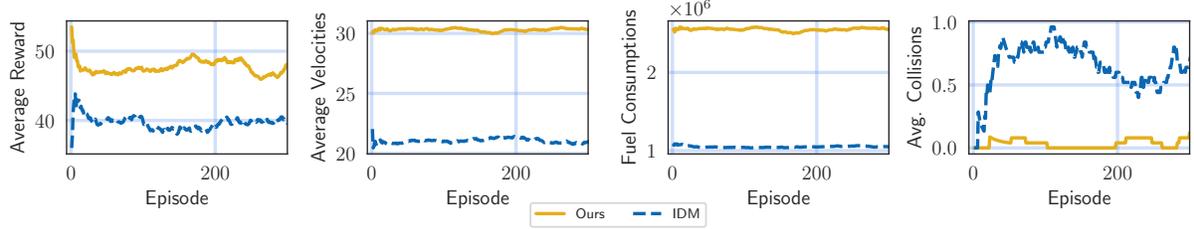


Fig. 3: Comparison of the proposed method against SUMO IDM in the evaluation phase.

performance metrics. In terms of average reward, our method shows a clear improvement over episodes. For average velocities, our model reaches and maintains a higher level, meaning vehicles move more smoothly and efficiently. Fuel consumption is slightly higher with our approach, probably due to higher velocity. Finally, the number of collisions drops significantly during training with our model, but IDM continues to have more frequent crashes. These results illustrate how carefully tailoring the information state of the ego vehicle to include only important nearby interplay dynamics can enhance training speed and evaluation performance.

VI. CONCLUSION

This paper presented a highway merging framework integrating partial attention into QMIX, where attention operates on two complementary levels: spatial attention identifies the relevant vehicles, while temporal attention extracts informative past states. Combined with a hybrid reward signal balancing global and individual objectives, this design yields safer and more efficient merging behavior. Simulation results in a SUMO-based environment confirm significant improvements in collision rate, average velocity, and overall reward over a standard driving baseline, with the trade-off of higher fuel consumption driven by increased speeds. Future work will extend the framework to multi-lane and mixed-autonomy settings where autonomous and human-driven vehicles coexist.

REFERENCES

- [1] K. Yang, X. Tang, S. Qiu, S. Jin, Z. Wei, and H. Wang, "Towards robust decision-making for autonomous driving on highway," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 9, pp. 11 251–11 263, 2023.
- [2] D. Chen et al., "Deep multi-agent reinforcement learning for highway on-ramp merging in mixed traffic," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 11, pp. 11 623–11 638, 2023.
- [3] W. Zhou, D. Chen, J. Yan, Z. Li, H. Yin, and W. Ge, "Multi-agent reinforcement learning for cooperative lane changing of connected and autonomous vehicles in mixed traffic," *Autonomous Intelligent Systems*, vol. 2, no. 1, p. 5, 2022.
- [4] J. Du, A. Yu, H. Zhou, Q. Jiang, and X. Bai, "Research on integrated control strategy for highway merging bottlenecks based on collaborative multi-agent reinforcement learning," *Applied Sciences*, vol. 15, no. 2, p. 836, 2025.
- [5] J. Chen et al., "Multi-agent deep reinforcement learning cooperative control model for autonomous vehicle merging into platoon in highway," *World Electric Vehicle Journal*, vol. 16, no. 4, p. 225, 2025.
- [6] Z. Chen, Y. Du, A. Jiang, and S. Miao, "Deep reinforcement learning algorithm based ramp merging decision model," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 239, no. 1, pp. 70–84, 2025.
- [7] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," *ICLR*, 2016.
- [8] Z. Li, T. Shang, and P. Xu, "Multi-modal attention perception for intelligent vehicle navigation using deep reinforcement learning," *IEEE Transactions on Intelligent Transportation Systems*, 2025.
- [9] T. Rashid, M. Samvelyan, C. S. De Witt, G. Farquhar, J. Foerster, and S. Whiteson, "Monotonic value function factorisation for deep multi-agent reinforcement learning," *Journal of Machine Learning Research*, vol. 21, no. 178, pp. 1–51, 2020.
- [10] P. A. Lopez et al., "Microscopic traffic simulation using SUMO," in *The 21st IEEE International Conference on Intelligent Transportation Systems*, IEEE, 2018.
- [11] C. Amato, G. Chowdhary, A. Geramifard, N. K. Üre, and M. J. Kochenderfer, "Decentralized control of partially observable Markov decision processes," in *52nd IEEE Conference on Decision and Control*, IEEE, 2013, pp. 2398–2405.
- [12] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.
- [13] A. Vaswani et al., "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.
- [14] Y. Chang et al., "A survey on evaluation of large language models," *ACM transactions on intelligent systems and technology*, vol. 15, no. 3, pp. 1–45, 2024.
- [15] M. Tan, "Multi-agent reinforcement learning: Independent vs. cooperative agents," in *Proceedings of the tenth international conference on machine learning*, 1993, pp. 330–337.
- [16] R. Sutton and A. Barto, *Reinforcement learning: An introduction*. MIT Press, 2018.
- [17] V. Mnih et al., "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.
- [18] K. Son, D. Kim, W. J. Kang, D. E. Hostallero, and Y. Yi, "Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning," in *International conference on machine learning*, PMLR, 2019, pp. 5887–5896.
- [19] J. L. Ba, J. R. Kiros, and G. E. Hinton, *Layer normalization*, 2016. arXiv: 1607.06450 [stat.ML].
- [20] J. Ansel et al., "PyTorch 2: Faster machine learning through dynamic Python bytecode transformation and graph compilation," in *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, ACM, 2024.
- [21] A. Wegener, M. Piórkowski, M. Raya, H. Hellbrück, S. Fischer, and J.-P. Hubaux, "TraCI: An interface for coupling road traffic and network simulators," in *Proceedings of the 11th communications and networking simulation symposium*, 2008, pp. 155–163.
- [22] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2019.
- [23] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Physical review E*, vol. 62, no. 2, p. 1805, 2000.