Multi-Agent System Techniques for Autonomous Vehicle Navigation and Traffic Management

Coordination, Control and Decision-Making in Intelligent Transportation Systems

Diana Brebeanu
Cheriton School of Computer Science
University of Waterloo
Waterloo, Ontario, Canada
dbrebean@uwaterloo.ca

ABSTRACT

As autonomous vehicles continue to evolve and integrate into real-world transportation systems, the demand for intelligent, adaptive, and fail-safe decision-making processes becomes increasingly important. This survey papers reviews recent multi-agent system techniques developed to enhance autonomous navigation and traffic coordination, with a particular focus on three challenges. The first two are from the perspective of the autonomous vehicle: merging into a lane and ensuring safe decision-making under uncertainty. The third challenge is from the perspective of the control system to coordinate traffic efficiently through an intelligent traffic light control system. The first section explores the task of lane merging, a scenario that requires both strategic coordination among vehicles and precise motion control. Models based on game theory, specifically two-player static games, are used to determine optimal merging sequences, simulating how vehicles negotiate right-of-way based on mixed-strategy Nash equilibria. These strategic decisions are then combined with bi-objective motion planning that leverages Pontryagin's Minimum Principle to minimize fuel consumption and travel time. A novel solution approach to traverse the Pareto front with a modified quicksort algorithm is introduced, allowing for efficient identification of optimal trade-offs between competing objectives. The second focus area centers on ensuring safe decision-making and addresses the challenges of operating under uncertain and adversarial conditions. Reinforcement learning is used to add robustness guarantees through the use of an adversarial training framework. The Robust Reinforcement Learning with Safety Guarantees (RRL-SG) model incorporates a learned adversary to simulate worst-case environmental perturbations and also adds a safety mask based on the Responsibility-Sensitive Safety (RSS) framework to eliminate high-risk actions. Furthermore, a Safe-Robust Markov Decision Process (MDP) formulation is introduced to iteratively refine policy and value functions using an actor-critic framework and robust policy improvement strategies. This dual analysis reveals important trade-offs and complementarities between coordination-centric and safety-centric methodologies. While merging strategies emphasize collaborative optimization in structured environments, robust RL approaches are designed to withstand uncertainty and maintain safety in unpredictable conditions. The third section focuses on intelligent traffic light control systems, formulating the problem of navigating through multiple intersections as a Semi-Markov game and utilizes the multi-agent attention double actor-critic (MAADAC) framework which combines the Double Actor-Critic reinforcement learning approach with graph attention networks. This allows agents to learn longer sequences of satisfactory decisions and was shown to reduce congestion through reducing the waiting time for high priority vehicles at intersections by 18.16%-38.14% compared to other state-of-the-art approaches. Together, these methodologies highlight the complex nature of autonomous navigation and demonstrate how integrating optimization, game theory, and machine learning can lead to more resilient and cooperative autonomous vehicle systems.

CCS CONCEPTS

- Computing methodologies
- Theory of Computation
- Information Systems
- Applied Computing

KEYWORDS

Autonomous vehicles, multi-agent systems, game theory, motion planning, reinforcement learning, decision-making under uncertainty, safe navigation, intelligent transportation systems, on-ramp merging, robust control

ACM Reference format:

Diana Brebeanu, 2025. Multi-Agent System Techniques for Autonomous Vehicle Navigation and Traffic Management: Coordination, Control and Decision-Making in Intelligent Transportation Systems. In *Proceedings of Waterloo ECE 493 (WATERLOO'25). ECE 493, Waterloo, ON, Canada, 2 pages.*

1 Introduction

As autonomous vehicles move from concept to reality, the intersection of autonomous driving, traffic control and connected vehicle networks presents important opportunities for researchers to tackle the challenges of ensuring safe traffic management and control of vehicle responses to changes in their environment. These have a fundamental impact on public safety through the co-operation of different vehicles to reduce potential accidents as well as on transportation efficiency through optimizing traffic flow to reduce congestion and carbon emissions. Achieving these results requires robust solutions to challenges such as vehicle communication, real-time decision making for joint benefits as well as fail-safe systems that prioritize safety. In this survey paper, we will focus on techniques from the perspective of the vehicles used for merging into a lane and ensuring safe decision-making processes centered around changing lanes. We then also look at techniques from the perspective of control systems to coordinate traffic, mainly the use of multi-agent systems in traffic light control.

1.1 Merging into a lane

In 2021, the paper "On-Ramp Merging Strategy for Connected and Automated Vehicles Based on Complete Information Static Game" by Min et al. [8] modelled the problem of merging into a lane by separating it into two concerns: merging sequence and motion planning. Merging sequence refers to scheduling the order in which cars leave the merging ramp and join the main road while motion planning is concerned with ensuring vehicles move in a safe way to ensure avoidance of collisions while also optimizing fuel consumption and traffic efficiency. The work done in "Cooperative Game Approach to Optimal Merging Sequence and On-Ramp Merging Control of Connected and Automated Vehicles" by Jing et al. [5] is also covered and focuses on modelling the merging sequence problem as a two-player complete information static game in which players simultaneously select their strategies. This is a competitive game in which the vehicles are competing for higher priority in the merging sequence. Each of the vehicles selects their strategy by calculating their mixed Nash strategy equilibrium.

For the motion planning, it is formulated as a bi-objective optimization problem based on Pontryagin's Minimum Principle. In this approach, the parameters of the cost function are determined using a search method called varying scale-grid, where the search scale progressively narrows down to refine the solution. This method of parameter search is not commonly found in existing literature. The solution to this optimization problem is then identified by employing a quicksort algorithm to traverse the Pareto front which can be defined as a set of solutions that are non-dominated and outperform other solutions in the entire solution space. This method of using the Pareto front for solution selection and the quicksort algorithm for optimization is also a novel contribution not typically seen in previous studies.

This two-pronged approach of using game theory for merging sequence and optimal control for motion planning demonstrates a well-integrated framework that mirrors the layered complexity of real-world merging behavior in connected and automated vehicle (CAV) systems. By decoupling the sequence decision from the physical control of vehicle movement, the researchers effectively reduced the computational complexity of the problem while preserving the interactions between strategic and dynamic elements. Notably, the mixed Nash equilibrium formulation captures the uncertainty and strategic interdependence between vehicles, aligning well with the decentralized nature of CAVs. Meanwhile, the bi-objective motion planning approach balances competing goals of safety and efficiency, and the use of Pareto front traversal with a quicksort-based search introduces a novel, computationally efficient method for identifying optimal trade-offs. Together, these methods reflect an advancement in both the modeling precision and the solution techniques for autonomous on-ramp merging, providing a foundation for future work to build more responsive and cooperative merging systems.

1.2 Ensuring Safe Decision-Making Processes

As previous sections focused on calculating decisions for vehicles in different traffic situations, it is also critical to ensure that these decisions consider the safety of the passengers with the highest importance. Reviewing the paper by He et al. [4], a new reinforcement learning technique is introduced to ensure collision safety through the development of an adversary model to simulate worst-case uncertainties. This is combined with an actor-critic algorithm to enable the agent to learn policies against adversarial perturbations to make trustworthy decisions and reduce collision likelihood. The framework for this is named RRL-SG (Robust Reinforcement Learning – Safety Guarantees).

Building on this foundation, the adversary model introduced by He et al. serves not just as a stress test but as a core training component to enhance the robustness of the decision-making process. By purposefully introducing perturbations that simulate rare or extreme driving conditions, such as sudden pedestrian crossings or unpredictable vehicle maneuvers, the model forces the reinforcement learning agent to account for edge-cases that traditional training might not take into account. This adversarial training ensures that the agent's learned policy is not only effective under ideal conditions but also resilient when exposed to unexpected hazards. The actor-critic framework further supports this by continuously adjusting both the policy and value estimations in response to these adversarial challenges, ultimately improving the system's ability to generalize and maintain safety across a wide range of traffic scenarios.

1.3 Coordinating high priority vehicles through intersections

Looking at traffic safety from the perspective of the intelligent traffic light control systems (ITLCS) in urban contexts, the paper "Multi-Agent Attention Double Actor-Critic Framework for Intelligent Traffic Light Control in Urban Scenarios With Hybrid Traffic" by Liu et al. [7] seeks to optimize the waiting time for high priority vehicles at multiple intersections. High priority vehicles can be described by examples such as school buses, ambulances and fire truck, snowplows and construction equipment trucks as well as public buses. Traditional traffic light systems use pre-defined fixed schemes for coordinating green and red-light timings. However, oftentimes these cannot meet the real-world demands of traffic flows made up of vehicles with varying priorities. Oftentimes, to ensure that high priority vehicles can pass through the intersection quickly, the traffic lights should allow enough time for a green light to let vehicles through to clear the path for the high priority vehicles so that they can pass through without slowing down. This requires adaptive time allocation for the different light phases of the intersection. Moreover, changing the light phases of an intersection will influence the nearby intersections and is important to be considered for facilitating the efficient flow of high priority traffic.

To address this, Liu et al. [7] proposed the Multi-Agent Attention Double Actor-Critic framework (MAADAC) which uses a Semi-Markov game to model the traffic light controls for multiple intersections and each intersection is treated as an agent. A Markov game which models the next state as a function of the current state and agent's actions. The Semi-Markov game builds upon this by changing the time spent in a state from being fixed to being variable. This means that the probability of transitioning to the next state also takes into account the time spent in a current state in addition to the action taken by the agent and current state. The MAADAC framework integrates the options framework with graph attention networks which allows the intersections to make long sequences of light phase decisions while also considering the influences that intersections' phases will have on each other.

2 Related Works

2.1 Merging into a lane

In 2017, Kang and Rahka [6] proposed a decision-making process for calculating a merging sequence based on non-cooperative game theory. In this paper, they discuss two possible states that the vehicles could be in: original state and lagging state when compared to the car in the other lane. The game played between the two cars determines which car gets to keep its original state and which car will lag the other one. A similar approach is used in the Min et al. paper [8] to determine the order of vehicles.

In 2019, Jing et al. [5] modelled merging sequence as a multi-player game which was then decomposed into multiple two-player games. This technique was then used in the paper by Min et al [8] and combined with motion planning.

For the motion planning problem**,** two primary approaches have been explored: centralized and decentralized systems. Centralized systems involve a central controller that manages the traffic flow. For instance, in 2015 Cao et al. [2] proposed a model where vehicles on the main road receive information from a central controller about vehicles on the ramp, allowing them to adjust their state to optimize the merging process. In contrast, decentralized systems rely on each vehicle making decisions based on the information it receives. In 2018, Z. Wang et al. [14] explored this approach in the context of on-ramp merging, where vehicles cooperate using Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) communication to adapt their speeds and positions, thereby improving traffic flow during merging scenarios.

2.2 Ensuring Safe Decision-Making Processes

A traditional and most popular technique used in decision-making is based on rule-based systems such as finite-state machines (FSM). This is generally simple to implement but relies on the expert knowledge of specialists which makes it difficult to design rules for vehicles to follow during more complicated traffic situations.

An alternative to this approach is reinforcement learning. As reinforcement learning is based around an agent interacting with the environment to learn optimal behaviour (policies) for similar future interactions, it is a useful tool for solving complex sequential decision-making problems that autonomous vehicles are tasked with. Ye et al. [15] published a paper outlining a reinforcement learning technique for determining policies for changing lanes and that these can be generalized to new environments. Moreover, reinforcement learning techniques have been used for determining optimal target speeds or speed patterns. However, most studies only consider at most one point of uncertainty which lowers the robustness of many of these approaches and their safety guarantees. The paper by He et al. [4] aims to consider policy robustness against multiple uncertainties.

2.3 Coordinating high priority vehicles through intersections

There have been many studies centered around various methods to dynamically adjust traffic lights for optimal traffic efficiency. In 2011, Oertel et al. [9] formalized a basic idea of adjusting the time of a green light that an isolated intersection has for a certain direction based on either the waiting vehicles' headways or delay times. The term headway defines the distance between consecutive vehicles. The delay time is the amount of time that a vehicle becomes delayed by while it is waiting at a red light. For the headway method, the intersection shows a green light for a certain direction and once the headways between vehicles surpasses a threshold, meaning that there are few vehicles moving in that direction anymore, it switches to a red light. The same process occurs for the delay time method once the delay time is lower than a certain threshold. This also takes into consideration a minimum and maximum time that a green light should be displayed for. This technique can be categorized as a rule-based method and there are other studies which also use manually designed objective functions to optimize traffic flow.

Exploiting the use of learning and taking into the account the influence that intersections have on each other, the 2019 study by Y. Wang et al. [13] introduced a distributed traffic light control framework which made use of a recurrent neural network to integrate historical traffic records, providing insight into the traffic conditions at certain times of the day for specific intersections, with a graph neural network to represent the spatial relationships between these intersections. These were then combined with the deep Q-learning method for making decisions for each intersection in a distributed way. Adding on the idea of differing priorities for certain vehicles, in 2017, Asaduzzaman et al. [1] developed a traffic control algorithm that prioritized emergency vehicles by adjusting the traffic light phase durations to reduce their travel time.

Liu et al. [7] combines this idea of relationships between intersections with prioritizing vehicles while adding to this by assigning varying priorities for vehicles with different importances.

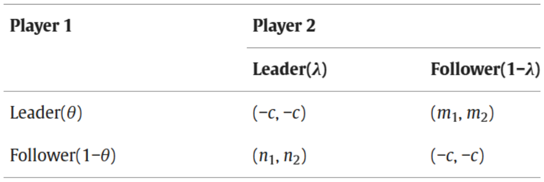
3 Methodology

3.1 Merging into a lane

The problem of computing how cars on a merging ramp will merge into the main road is split into two parts. The first is the merging sequence problem, which is essentially a scheduling problem. This is concerned with determining the order of the cars while merging. This means comparing a car in the main road with a car on the merging ramp, determining which car will go in front of the other after having merged. The second part is motion planning which deals with the actual motion of vehicles to avoid collisions, optimize fuel consumption and enhance traffic flow.

3.1.1 Merging Sequence. The merging environment is modelled as a ramp that merges into a main road. This area is split into two sections: the control area before the merging ramp intersects with the main road (called the merging point) and the merging area where the ramp and main road unify. The control area is where the vehicles are meant to adjust their speeds according to a central controller before merging. In modelling the problem, several simplifying assumptions are made including that there are no time delays for sending signals between the centralized controller and the vehicles, no overtaking is allowed, and we are only looking at situations where there is one lane on the main road and one lane on the merging ramp. No overtaking means that no cars will speed up to pass a vehicle, they will only decide whether to slow down to lag. This will also help ensure safety.

The merging sequence is modelled as a two-player game. When a vehicle V_m in the merging lane enters the merging section, a game begins between itself and the first vehicle in the main road V_i which will determine who will be in the lead after merging. After this, another game will start between that the vehicle that lost the game (is now behind the leader) and the next vehicle in the other ramp.



Player 1	Player 2	
	Leader(λ)	**Follower($1-\lambda$)**
Leader(θ)	$(-c, -c)$	(m_1, m_2)
Follower($1-\theta$)	(n_1, n_2)	$(-c, -c)$

Figure 1: Payoff matrix for a two-player merging sequence game from [8]

The payoff matrix is shown in Figure 1 where c represents the cost of a collision between vehicles and is defined as described in (1)

$$c = \frac{1}{\left\|\chi_1(t_g) - \chi_2(t_g)\right\|} \qquad (1)$$

The χ function describes the state of a player. The cost c is then greater if the states of each player are similar. m_1, m_2, n_1, n_2 are the payoffs of each player when they co-operate and are related to fuel consumption and travel time which will be explained in the motion planning section.

Solving for the mixed-strategy Nash equilibrium of the payoff matrix gives the following best response probability distributions in (2) for each player resulting in the mixed Nash equilibrium as $[(\theta^*, 1-\theta^*), (\lambda^*, 1-\lambda^*)]$

$$\theta^* = \frac{n_2 + c}{2c + m_2 + n_2} \text{ and } \lambda^* = \frac{m_1 + c}{2c + m_1 + n_1} \qquad (2)$$

Because the equilibrium is a pure strategy, not a mixed one, if the player's states are similar such that $\chi_1(t_g) \to \chi_2(t_g)$ which causes $c \to \infty$, then $\theta^* \to \frac{1}{2}$ and $\lambda^* \to \frac{1}{2}$ which means both players could choose to be the follower or leader in the merging sequence. To avoid collisions, a convention is set that Player 1 will become the leader and Player 2 will become the follower.

3.1.2 Motion Planning. Building on top of the work by Jing et al [5], the cost function for each vehicle V_i is defined as

$$J_i = \frac{1}{2}\int_{t_i^0}^{t_i^f} w_1 a_i^2(t)dt + w_2 T_i \qquad (3)$$

Where $a_i(t)$ represents the acceleration of vehicle V_i at moment t and T_i represents the travel time $(T_i = t_i^f - t_i^0)$. The first term in the cost function relates to fuel consumption while the second term relates to travel time with the coefficients w_1 and w_2 representing their weights.

Motion planning is then formulated as a global optimization problem to find the minimum cost by minimizing over possible accelerations with added limits on minimum and maximum possible values of accelerations and velocity.

$$\min_{a_i} J_i \qquad (4)$$

$$a_{min} \leq a_i(t) \leq a_{max},\ 0 \leq v_i(t) \leq v_{max},\ \forall t \in \{t_i^0, t_i^f\}$$

Pontyagrin's principle is then used to calculate the relationship between a vehicle's position, velocity and acceleration used during merging from this optimization problem dependent on the cost function. The payoffs from the matrix in Figure 1 are then derived from these values.

The two parameters w_1 and w_2 for the cost function are determined through running a varying-scale grid search to find an unbiased Pareto solution. This means a solution which is nondominated, any deviation would deteriorate at least one of the objectives: fuel consumption and travel time.

The solution is found placing initial limits l_1 and l_2 on w_1 and w_2 respectively and initializing search step lengths σ_1 and σ_2 along with scaling factors α_1 and α_2. Then the fuel consumption and travel time in (5) and (6) respectively are calculated based on the values of w_1 and w_2 (substituting in W in the equation). Note that $f_i(t)$ calculates the fuel consumption based on the previously calculated velocity and acceleration of the car.

$$F = \sum_i^W \sum_j^N f_i(j) \qquad (5)$$

$$T = \sum_i^W T_i \qquad (6)$$

The Pareto front is then found using modified quicksort and the new values for F and T are then set to w_1 and w_2. These are then used to calculate the new limits l_1 and l_2 and new search step lengths σ_1 and σ_2. This process is then repeated until there are no longer significant changes in the step values.

3.2 Ensuring Safe Decision-Making Processes

To train an agent to anticipate multiple uncertainties, He et al. [4] train an adversarial agent online to model worst-case uncertainties through generating optimal perturbations of the observed states that a vehicle is in and the characteristics of the environment it interacts with. Furthermore, a safety mask is added to ensure collision safety which transforms the probability of selecting an unsafe decision to zero. This is then used with an adversarial robust actor algorithm which allows the agent to learn robust policies against these perturbations.

Pertaining to the agent's own characteristics, the state of an agent is designed to have 15 dimensions which include its own velocity, acceleration and lane index as well as the relative distance and velocity of the six nearest vehicles in the same or adjacent lanes. The action space is discrete and contains the following possible actions: changing lanes to the left, changing lanes to the right, maintaining the current state, accelerating at a fixed rate of 1.47 m/s^2 and decelerating at -2.00 m/s^2.

3.1.1 Adversary Model. The optimal adversarial perturbations observed states, and environmental dynamics are represented as Δ_o^* and Δ_d^* in the form of probability distributions. The adversarial model takes in the state s of the agent and outputs these two perturbations. Δ_o^* is meant to add noise to the agent's observations such that it causes the worst-case confusion in the agent, expressed as maximizing the average variation distance on perturbed policies. Δ_d^* is then meant to minimize the expected return of the agent through causing the environment to behave in a way that causes worst-case rewards for the agent.

To quantify policy variation under adversarial observation noise, the model employs the Jensen–Shannon (JS) divergence, a symmetric and bounded variant of the Kullback–Leibler (KL) divergence. The JS divergence measures the difference between the original policy and the perturbed policy, encapsulated in an objective function J_o, which accounts for the change in action distributions due to observation perturbations.

To perturb the environmental dynamics, the agent's expected return is estimated by the Q function $Q_s^{\pi}(s)$ which takes in the state of the agent and assumes the action follows the policy π. This is then captured by the objective function

$$J_d(s, Q^{\pi}, \Delta_d) = \Delta_d Q^{\pi}(s) \qquad (1)$$

These two objective functions are then combined and weighted to give (2)

$$J_{\Delta}(s, \pi, Q^{\pi}, \Delta) = (\alpha - 1)J_o(s, \pi, \Delta_o) + \alpha J_d(s, Q^{\pi}, \Delta_d) \; (2)$$

This is then used in the following optimization problem to get the optimal parameters of the adversary model. The optimization has been simplified using the hyperbolic tangent and SoftMax functions on the perturbations

$$\theta * \in argmin_{\theta} E[J_{\Delta}(s, \pi, Q^{\pi};\ \theta)] \quad (3)$$

3.1.2 Safety Mask. To ensure the collision safety of autonomous vehicles, a safety mask based on the Responsibility-Sensitive Safety (RSS) framework is developed using a jerk-bounded model. This model, derived from Intel, accounts for a realistic braking profile where a vehicle decelerates with a bounded jerk until a minimum deceleration is reached, followed by constant deceleration until a full stop. The minimum longitudinal safe distance D_{min}^{RSS} is computed accordingly considering vehicle speeds, accelerations, and braking dynamics.

This approach is then extended to lateral maneuvers such as lane changes, where a minimum lateral safety distance is introduced as a scaled version of D_{min}^{RSS} using a safety coefficient. The safety mask modifies decision-making probabilities by assigning negative infinite rewards to actions that violate these longitudinal or lateral safety distances. For instance, if the distance to a vehicle in the target lane is insufficient, lane-change and acceleration actions are suppressed. This masking technique combines reinforcement learning with rule-based filtering of unsafe maneuvers, enhancing decision safety without extensive retraining.

3.1.3 Adversarial Robust Actor-Critic Algorithm. A Markov Decision Process (MDP) is used as the basis for finding the optimal policy of the agent. In this case, the standard MDP is extended to model the behaviour of the agent under the adversarial perturbations and safety mask constraints. A new Safe-Robust MDP is defined as a max-min problem in (4) where T is the last time step and $\beta > 0$ is a trade-off coefficient:

$$max_{\pi} min_{\Delta}\, E[\Sigma_{t=0}^{\mathrm{T}} \gamma^t r(s_t, a_t) + \beta J_{\Delta}(s, \pi, Q^{\pi}, \Delta)] \qquad (4)$$

The reward function r was designed to represent driving safety, passenger comfort and travel efficiency and as opposed to Min et al. [8], fuel consumption was not taken into consideration. The reward function rewarded the agent for driving at high speeds while also penalizing the agent if any maneuvers caused a collision or performed risky movements such as high-speed lane changes.

Finding the optimal policy is done in two steps, safe-robust policy evaluation and robust policy improvement which are iterated until convergence.

3.1.3.1 Safe-Robust Policy Evaluation. The policy evaluation phase estimates the expected return of a fixed policy under environmental uncertainty. To account for possible adversarial perturbations, the standard Bellman backup operator $\Psi^{\pi,\Delta}$ is used to capture the effect of such perturbations on future value estimates.

$$\Psi^{\pi,\Delta} Q^{\pi}(s_t) = r_a(s_t, a_t) + \gamma \pi(s_{t+1}) Q^{\pi}(s_{t+1}) \qquad (5)$$

This includes in an augmented reward that reflects both the reward and the influence of uncertainty and action-value function $Q^{\pi}(\cdot)$ which estimates the expected return based on the state and action of the agent when it follows policy π.

$$r_a = \mathrm{r}(s_t, a_t) + \gamma \beta J_{\Delta}(\cdot) \qquad (6)$$

To perform policy evaluation effectively, the authors employ two separate critic networks, each representing a parameterized action-value function. These networks are trained using a loss function that measures the discrepancy between predicted and target value estimates. The target is computed conservatively by taking the minimum value across both critic networks, which helps mitigate overestimation bias commonly observed in value-based reinforcement learning. The critic networks are then updated by minimizing the squared Bellman error using stochastic gradient descent. To stabilize training further, the target networks are maintained using Polyak averaging, which softly updates target parameters as a weighted average of current and previous values. This approach enhances the safety and robustness of policy evaluation by explicitly modeling uncertainty and leveraging conservative value estimation strategies during training.

3.1.3.2 Robust Policy Improvement. This step is concerned with optimizing the policy π given the action-value function $Q^{\pi}(\cdot)$ under the adversarial perturbations. This is represented as a max-min game

$$max_{\pi}\ min_{\Delta}\ E[J(\pi, \Delta)] \quad (7)$$

Where $J(\cdot)$ corresponds to the objective function from (4) such that $J(\pi, \Delta) = \pi(s) Q^{\pi}(s) + \beta J_{\Delta}(s, \pi, Q^{\pi}, \Delta)$ and recalling that J_{Δ} comes from (2).

The optimal policy π^* and optimal adversarial perturbation Δ^* are then found by

1. Fixing an arbitrary policy π
2. Solving for Δ^* according to the optimization problem in (8)
3. Learning π^* with Δ^* using according to (9)

$$\Delta^* = arg\ min_{\Delta}\ E[J(\pi, \Delta)] \qquad (8)$$

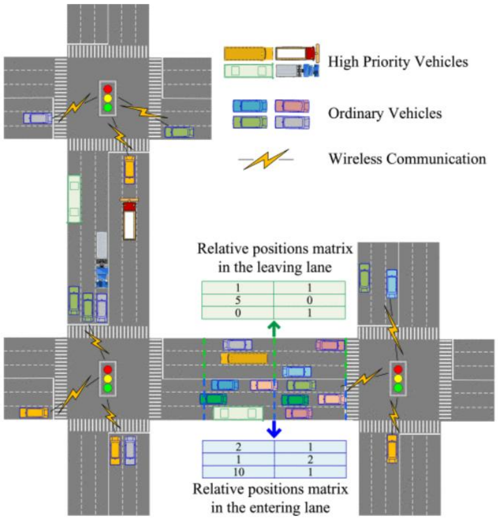
$$\pi^* = arg\ max_{\pi}\ E[J(\pi, \Delta^*)] \qquad (9)$$

We can see that this represents a zero-sum game and based on the derived results, a convergence of the policy improvement is guaranteed.

3.3 Coordinating high priority vehicles through intersections

Liu et al. [7] set up the problem of optimizing the traffic flow for vehicles with differing priorities by assigning them each with importance weights according to the category they are in (e.g. emergency vehicles will have a higher importance weight than private vehicles). For each individual intersection, it will be configured with traffic light control scheme to minimize the weighted waiting time of the vehicles in the intersection, where the weights correspond to the vehicles’ importances.

3.3.1 Problem set up and Definitions. The intersection’s traffic conditions are also represented by dividing each road segment into a series of grids. A matrix is used where each cell represents a grid segment and contains the importance values of the vehicles in the grid segment. The lanes leaving the intersection would contribute to the intersection’s gains as they represent the vehicles that were able to pass through. Conversely, the lanes entering the intersection contribute to the decision-making costs for the intersection. Because of this, we store separate matrices for the entering and leaving lanes.



Relative positions matrix in the leaving lane

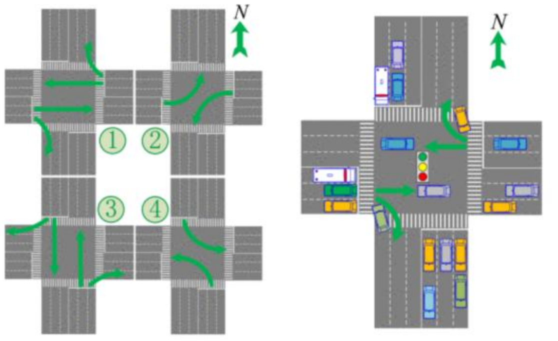
1	1
5	0
0	1

2	1
1	2
10	1

Relative positions matrix in the entering lane

Figure 2: Visualization of vehicles in the grid segments of the road represented as matrices [7]

The different phases that an intersection could be in are summarized with 4 different light phases seen in Figure 3. An example of a phase is when vehicles that are driving straight through the intersection or turning right are able to pass through. A different phase would be when only left-turning vehicles are passing through. These two phases are then applied to the perpendicular directions of traffic, totalling in 4 different phases.



(a) The four-phase assignment scheme.

(b) A snapshot of the traffic conditions when phase 1 is active.

Figure 3: Visualization of the four possible phases that an intersection can be in [7]

Looking at multiple intersections, the decision-making process for multiple intersections is modelled as a Semi-Markov game. It uses the options framework which defines possible options that the agent can take as they can differ depending on the state and actions taken. The Semi-Markov game is defined with the following components at each time step *t*:

Agent: Each intersection i **is** an agent i ϵ I where I $= \{i \mid i = 1, \ldots, N\}$

State: $s_t \, \epsilon \, S$ contains the number of agents, each of their phases and position and priority information of the vehicles for each agent

Observation: $o_i \, \epsilon \, O$ consists of the matrices describing the positions of the vehicles with different weights of priority

Option: $\omega_{i,t} \, \epsilon \, \Omega$ which is equal the triple $(I_\omega, \pi_\omega, \beta_\omega)$ where I_ω is a set of agents, π_ω is the low-level policy and β_ω is a termination function which terminates the current option if $\beta_\omega = 1$

Action: $a_i \, \epsilon \, A_i$ denotes the action from agent i’s action space and represents the phase chosen for the next period of time t_p

High-level Policy: given the previous option $\omega_{i,t-1}$, the current observation $o_{i,t}$, agent i’s high-level policy $\pi^H_{i,t}$ specifies a probability for taking an option $\omega_{i,t}$ as: $\pi^H_{i,t}(\omega_{i,t} | \omega_{i,t-1}, o_{i,t})$

Low-level Policy: given the current option $\omega_{i,t}$, the current observation $o_{i,t}$, agent i’s low-level policy $\pi^L_{i,t}$ specifies a probability for taking an action $a_{i,t}$ as: $\pi^L_{i,t}(a_{i,t} | o_{i,t}, \omega_{i,t})$

Transition Probability in the High-level module: given a state s_t, the previous join option $\boldsymbol{\omega_{t-1}} = (\omega_{1,t-1}, \ldots, \omega_{N,t-1})$ and the current joint option $\boldsymbol{\omega_t} = (\omega_{1,t}, \ldots, \omega_{N,t})$, the transition probability is noted as $P^H(\boldsymbol{\omega_t}, s_{t+1} | \boldsymbol{\omega_{t-1}}, s_t, \boldsymbol{\omega_t})$

Transition Probability in the Low-level module: given a state s_t , the previous join option $\boldsymbol{\omega_t} = (\omega_{1,t-1}, \ldots, \omega_{N,t-1})$ and the joint action $\boldsymbol{a_t} = (a_{1,t}, \dots, a_{N,t})$, the transition probability is noted as $P^L(s_{t+1}, \boldsymbol{\omega_{t+1}} | s_t, \boldsymbol{\omega_t}, \boldsymbol{a_t})$

Reward: the objective of each agent is to minimize the weighted waiting time of all vehicles around the intersection which is defined as $r_{i,t} = \eta \Sigma_v \lambda_v \cdot (w_{i,v,t-1} - w_{i,v,t})$ where λ_v is the importance weight of vehicle v, $w_{i,v,t}$ is the accumulated waiting time for a vehicle v around agent i and η is a small constant

3.3.2 MAADAC Framework. The MAADAC framework consists of two types of modules: high-level module which is responsible for selecting the suitable option to guide the agent through a long sequence of decisions and the low-level module which is responsible for switching to a satisfactory phase. Graph Attention Networks (GAT) are used to connect neighbouring intersections and represent their influence on one another. Intuitively, this can be thought of as each intersection being a vertex in a directed graph and each edge representing a road that connects the intersections. For more details on GATs, see [12].

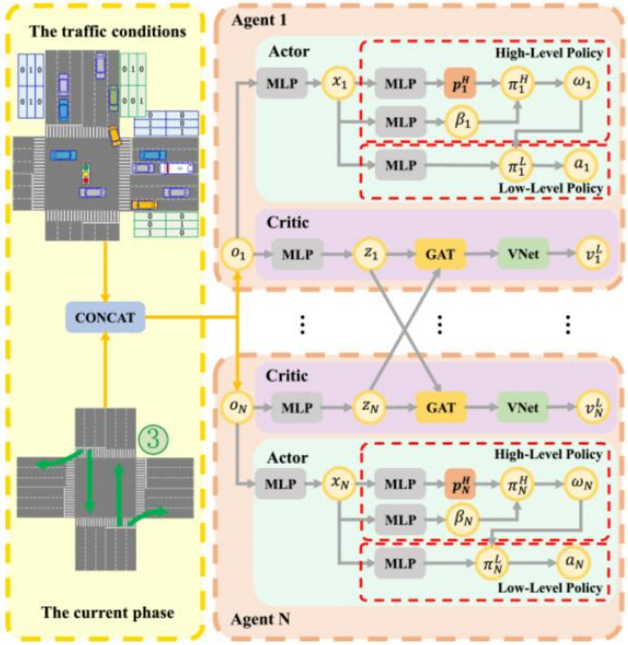


Figure 4: The MAADAC framework taking in the current traffic state [7]

3.3.3 Actor-Critic Design. The Actor-Critic framework encompasses two networks where the actor learns two policies (one for the high and one for the low-level module) to propose an action, and the critic evaluates the quality of this action through a value function to calculate a value for each module.

From Figure 4 we see that an agent i’s observation is fed into multi-layer perceptron (MLP) layer of the actor to obtain a latent representation $x_{i,t}$. This is then used to get a probability distribution $p_{i,t}^H$ over agent i’s option space as well as a termination condition $\beta_{w_{i,t}}$. These two are used to determine the high-level policy $\pi_{i,t}^H$ (where II is the indicator variable):

$$\pi_{i,t}^H(\omega_{i,t} | \omega_{i,t-1}, o_{i,t}) = \beta_{w_{i,t-1}} p_{i,t}^H(w_{i,t} | o_{i,t}) + (1 - \beta_{w_{i,t-1}}) II_{w_{i,t-1}=w_{i,t*}} \quad (10)$$

The high-level then uses this policy to sample an option $\omega_{i,t}$. The low-level policy uses $x_{i,t}$ to get probability distributions over the agent i’s action space. It then takes the sampled option $\omega_{i,t}$ from the high-level module and uses this to choose one probability distribution to be the agent’s low-level policy $\pi_{i,t}^L$. From this it generates the action taken by the agent.

The Critic’s goal is to generate a state-value which is used to update the policies in the high and low-level modules. The Critic does this by taking in the observation and uses its own network to represent it as $z_{i,t}$. As this is being done for every agent, each $agent\ j's\ z_{j,t}$ is passed into the GAT network to be able to incorporate the mutual influences that agents have on each other. For each agent i, the output from the GAT layers is concatenated with the original $z_{i,t}$ representation, denoted as $(z_{i,t}, z_{i,t}^1, \dots, z_{i,t}^C)$. This is fed into a VNet to obtain the low-level module’s state value $v_{i,t}^L$. This is then used to calculate the state value of the high-level module for the current option and future observation as follows:

$$v_i^H(\omega_{i,t}, o_{i,t+1}) = \Sigma_{\omega_{i,t+1}} \pi_i^H(\omega_{i,t+1}) v_i^L(o_{i,t+1}, w_{i,t+1})$$

These values $v_{i,t}^L$ and $v_{i,t}^H$ are then used to update the high-level and low-level policies.

3.3.4 Training. The MAADAC framework uses centralized training with a decentralized execution, and it uses Proximal Policy Optimization (PPO) [10] to train both high and low-level policies. A centralized critic is trained for each agent during the training process which allows each agent to use extra information of other agents during the training process. However, during the execution process, each agent’s actor only has access to its local information. The training happens in two stages. The first stage trains the low-level policy through minimizing the low-level critic and actor loss functions. The second stage trains the high-level policy as well as the termination conditions through minimizing the high-level critic and actor loss functions.

The first stage keeps the high-level policy and terminal conditions fixed while the low-level policy returns an action which receives a reward from the environment. At the same time, the critic calculates the low-level state-value. These values are stored in a replay buffer which is then used to create the loss function that is minimized by the critic during its training. According to PPO, the actor is trained through minimizing a clipped objective function that uses importance sampling [7] to limit large policy updates, improving stability. Similarly, in the second stage, the low-level policy is fixed, and the high-level policy and terminal conditions are optimized through the same process. The framework alternates between these two stages for each agent and each time step over the training period.

4 Experimental Results

4.1 Merging into a lane

To explore the impact of varying parameters, the simulation focused on a control area with 10 vehicles, a control zone that was 40 meters long, and a 30-meter merging area. Vehicle speeds were randomly initialized with a normal distribution (mean of 15 m/s), and all vehicles were controlled to merge at a constant speed of 13.4 m/s. Constraints for acceleration and velocity were set within limits of [−3, 3] m/s^2 and [0, 30] m/s, respectively.

The initial step size for the search was then set at 0.05, with a scaling factor of 0.2. The search for the coefficients, w_1 and w_2, was conducted across a grid range of [0, 1], with surrogate objectives of fuel consumption and total travel time being equally weighted. After identifying potential Pareto-optimal solutions, the best solution was selected based on minimal bias, yielding coefficient values of $w_1 = 0.30$ and $w_2 = 0.80$. The search was refined by narrowing the range to [0.25, 0.35] for w_1 and [0.75, 0.85] for w_2, with a finer precision of 0.01. The final optimal coefficient values were $w_1 = 0.32$ and $w_2 = 0.79$, which were selected after the search precision requirement was met.

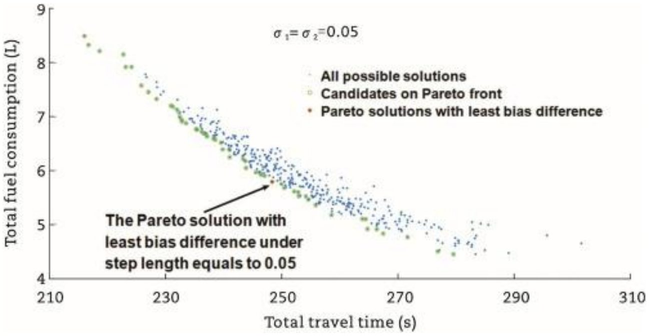


Figure 5: Simulation results from grid search under a 0.05 step length [4]

The experiments showed that all constraints were able to be satisfied, and the vehicles were all able to cooperatively merge. Moreover, the centralized controller ensured that vehicles passed the merging area in a coordinated manner, maintaining a constant distance of 30 meters between vehicles, ensuring safety. The vehicles also adhered to a consistent time headway, calculated as the length of the merging area divided by the merging speed (M/v_m), ensuring smooth and safe merging.

4.2 Ensuring Safe Decision-Making Processes

To evaluate the effectiveness and robustness of the proposed RRL-SG framework for safe decision-making, comparisons were conducted against several state-of-the-art RL baselines. These included dueling double deep Q-networks (D3QN) as a representative Q-learning algorithm, proximal policy optimization (PPO) for on-policy learning, soft actor-critic (SAC) for off-policy learning, and observation adversarial reinforcement learning (OARL) for robust RL.

Performance was assessed using a variety of metrics tailored to reflect both efficiency and safety in autonomous driving. The expected return was used as the primary indicator of overall policy performance. Additionally, average running speed and number of collisions were recorded to evaluate travel efficiency and traffic safety, respectively. To assess robustness, the extent of policy change under adversarial perturbations was measured; smaller changes indicated stronger resistance to attacks. For the on-ramp merging task, an additional metric—the merging success rate—was used, defined as the rate at which a vehicle successfully merged into the main lane without collisions.

Experiments were carried out in the Simulation of Urban MObility (SUMO) environment. Agents were trained and tested across two scenarios: a highway setting and an on-ramp merging scenario. In the highway scenario, traffic density was varied by altering the vehicle spawn probability P across low (0.06), normal (0.12), and high (0.24) levels. Each agent was trained under normal traffic density and tested under all three densities. For each algorithm, five independent training runs were performed using different random seeds. Evaluations were averaged over 100 test episodes, with ten episodes sampled per evaluation to account for stochastic variability in traffic flow.

To evaluate robustness, trained agents were tested under adversarial observational attacks generated by a learned adversary model. During these tests, the input states to the agent were perturbed to simulate the impact of adversarial environmental noise, thus enabling a realistic assessment of policy stability and resilience. The highway scenario was used both for training and testing, while the on-ramp merging scenario was exclusively used for testing. The focus was on assessing the generalization of trained agents to this unseen task, specifically evaluating merging behavior in terms of safety and success under realistic traffic dynamics.

Quantitatively, in normal-density traffic without adversarial interference, RRL-SG achieved return improvements of approximately 22.31%, 7.22%, 10.34%, and 1.97% over D3QN, PPO, SAC, and OARL, respectively. These gains became even more pronounced in high-density environments, where returns improved by up to 78.63%, 47.41%, 25.45%, and 13.84%, respectively. Under adversarial conditions in high-density traffic, RRL-SG delivered dramatic return gains of 7669.57%, 2666.25%, 511.57%, and 8.99% over the same baselines.

Overall, the proposed framework demonstrates a significant improvements over state-of-the-art baselines across various metrics related to safety, robustness, and expected return. Compared to D3QN, PPO, SAC, and OARL, the agent consistently achieved superior performance in both adversarial and non-adversarial settings. These improvements are attributed to the integration of an RSS-based safety mask, which restricts the action space to safe regions, thereby mitigating the risk of collisions and enhancing learning efficiency by reducing unnecessary exploration.

4.3 Coordinating high priority vehicles through intersections

In the Liu et al. paper [7], evaluation of the MAADAC framework was done through simulation of two real-world urban road networks, Suzhou Industrial Park and Daxing District in Beijing, built using the SUMO environment. The simulation was set up with 16 intersections, a vehicle arrival rate of 1.2 vehicles/s with 10% of the arriving vehicles being high priority vehicles.

4.3.1 Performance comparison against baseline methods. The framework was compared against the baseline methods: MAADAC combined with Advantage Actor-Critic (MAADAC+A2C) [16], Multi-Agent Advantage Actor-Critic, MADAC [3] and Rule-based (using delay time as described by Oertel et al. [9]).

The evaluation is concerned with verifying how much the waiting times at intersections of vehicles with different priorities is reduced. This is described with the following three metrics, all in seconds:

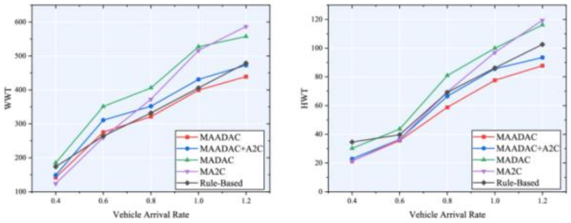
Weighted weighing time (WWT): the average weighted waiting time for all vehicles, directly related to the reward function.

High priority vehicles' waiting time (HWT): the average waiting time of the high priority vehicles.

Waiting time (WT): the average waiting time for all vehicles.

Then the ratio of HWT to WT (HWTR) is used to analyze how much priority is given to high priority vehicles with a lower ratio meaning a high focus on high priority vehicles.

As we can see in Figure 6, the WWT and HWT are the lowest for the MAADAC framework in the Beijing simulation which shows that the framework was able to respond to the presence of high priority vehicles. Overall, it improved the waiting time by 18.16% to 38.14% compared to the baselines. MAADAC worked much better than the simple Rule-based framework as the Rule-based framework could not grasp the significance of different traffic compositions and high priority vehicles. Moreover, it outperformed the MAADAC+A2C and MA2C frameworks, showing that the options framework combined with GATs works better on the framework.

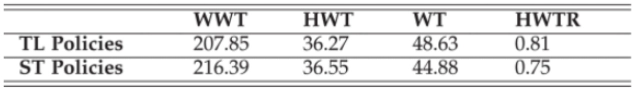


(a) The WWT versus the vehicle arrival rate.

(b) The HWT versus the vehicle arrival rate.

Figure 6: Performance comparison of MAADAC framework to other baseline frameworks in the Beijing simulation [7]

4.3.2 Performance evaluation with transfer learning. Similar results were found on the Suzhou simulation. A more interesting finding was the success of transfer learning. The agents that were trained on the Beijing simulation were then applied to the Suzhou scenario which created a transfer learning setting. The transfer learning policies (TL Policies) in this setting were learned on the Suzhou simulation using the agents' existing knowledge from the Beijing simulation. These were then compared with the single task policies (ST Policies) that came from agents that were only trained on the Suzhou simulation. The TL Policies resulted in lower WWT and HWT than the ST Policies, showing that transfer learning from the MAADAC framework allowed agents to adapt to new traffic conditions more efficiently.



	WWT	HWT	WT	HWTR
TL Policies	207.85	36.27	48.63	0.81
ST Policies	216.39	36.55	44.88	0.75

Figure 7: Performance comparison of TL Policies to ST Policies in the Suzhou simulation [7]

5 Conclusion

As autonomous vehicles continue to integrate into existing traffic systems, the challenge of safely managing their interactions with traditional vehicles becomes increasingly critical. This survey paper highlights key techniques in addressing specific traffic scenarios, particularly lane merging and ensuring safe decision-making processes. By utilizing concepts such as game theory, optimization, and advanced communication systems, researchers have made significant strides in designing systems that not only improve safety but also enhance overall traffic efficiency. The methodologies discussed, such as the two-player game models for merging and the optimization of fuel consumption and travel time, provide valuable insights into how self-driving cars can interact with each other and their environment. In particular, the merging strategies illustrate a dual-layered approach: strategic planning through Nash equilibria to determine merging order, and dynamic motion planning using Pontryagin's Minimum Principle to minimize travel time and fuel consumption while avoiding collisions. The potential benefits of these techniques extend beyond just collision avoidance to include smoother traffic flow, reduced congestion, and a positive impact on environmental sustainability.

In contrast, the techniques for ensuring safe decision-making focus on robustness under uncertainty and the handling of adversarial conditions. The introduction of an adversarial reinforcement learning framework (RRL-SG) extends traditional policy learning by explicitly accounting for worst-case scenarios using adversary-generated perturbations. Additionally, safety is proactively embedded using a safety mask based on the Responsibility-Sensitive Safety (RSS) framework, which guarantees collision avoidance by eliminating unsafe decisions from the agent's action space. This approach differs from merging models by prioritizing policy robustness and system reliability in unpredictable environments over traffic flow optimization.

Looking at optimizing traffic flow through the perspective of the traffic light control systems, we see there are numerous frameworks that consider important aspects of traffic situations such as vehicle delay time, influence of intersections on one another and differing vehicle priorities. The MAADAC framework proposed by Liu et al. [7] takes a novel approach with its use of the options framework integrated with GATs. Through simulations, it was shown that this framework supports and significantly improves the flow of traffic taking into consideration all the previous aspects when compared to other baseline models. Moreover, as the transfer learning showed promising results, the framework can be adapted to new traffic scenarios.

Together, these techniques work to address the complex nature of autonomous vehicle navigation. While merging solutions excel at structured interaction and efficiency under known conditions, robust decision-making frameworks ensure system resilience and safety in complex, uncertain environments. Meanwhile, intelligent traffic light control systems take into account the mutual influences that intersections, traffic conditions and varying priorities have on each other. The potential benefits of all domains extend beyond collision avoidance to include smoother traffic flow, reduced congestion, and increased trustworthiness.

However, while these solutions show promise, they also highlight the complexity of achieving fully coordinated autonomous driving systems. The integration of real-time decision-making, vehicle-to-vehicle and vehicle-to-infrastructure communication, and fail-safe mechanisms remains an ongoing challenge. Future research will need to refine these techniques and explore new ones to ensure that autonomous vehicles can seamlessly interact within human-driven traffic systems, ultimately paving the way for safer, more efficient roads.

ACKNOWLEDGMENTS

I would like to thank Professor Zahedi for his valuable instruction on the fundamentals of game theory and multi-agent systems, as well as for his thoughtful responses to questions on these topics which aided in writing this paper.

REFERENCES

[1] M. Asaduzzaman and K. Vidyasankar, "A Priority Algorithm to Control the Traffic Signal for Emergency Vehicles," 2017 IEEE 86th Vehicular Technology Conference (VTC-Fall), Toronto, ON, Canada, 2017, pp. 1-7, doi: 10.1109/VTCFall.2017.8288364.

[2] W. Cao, M. Mukai, T. Kawabe, H. Nishira, and N. Fujiki, "Cooperative vehicle path generation during merging using model predictive control with real-time optimization," Control Engineering Practice, vol. 34, pp. 98–105, Jan. 2015, doi: 10.1016/j.conengprac.2014.10.005.

[3] T. Chu, J. Wang, L. Codecà, and Z. Li, "Multi-agent deep reinforcement learning for large-scale traffic signal control," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 3, pp. 1086–1095, Mar. 2020.

[4] X. He, W. Huang, C. Lv, "Toward Trustworthy Decision-Making for Autonomous Vehicles: A Robust Reinforcement Learning Approach with Safety Guarantees," Engineering, vol. 33, pp. 77–89, Feb. 2024, doi: 10.1016/j.eng.2023.10.005.

[5] S. Jing, F. Hui, X. Zhao, J. Rios-Torres, and A. J. Khattak, "Cooperative Game Approach to Optimal Merging Sequence and on-Ramp Merging Control of Connected and Automated Vehicles," IEEE Transactions on Intelligent Transportation Systems, vol. 20, no. 11, pp. 4234–4244, Nov. 2019, doi: 10.1109/TITS.2019.2925871.

[6] K. Kang and H. A. Rakha, "Game Theoretical Approach to Model Decision Making for Merging Maneuvers at Freeway On-Ramps," Transportation Research Record, vol. 2623, no. 1, pp. 19–28, 2017, doi: 10.3141/2623-03.

[7] B. Liu, W. Han, E. Wang, S. Xiong, L. Wu, Q. Wang, "Multi-Agent Attention Double Actor-Critic Framework for Intelligent Traffic Light Control in Urban Scenarios with Hybrid Traffic," IEEE Transactions on Mobile Computing, vol. 23, no. 1, pp. 660–672, Jan. 2024, doi: 10.1109/tmc.2022.3233879.

[8] H. Min, Y. Fang, X. Wu, G. Wu, X. Zhao, "On-Ramp Merging Strategy for Connected and Automated Vehicles Based on Complete Information Static Game," Journal of Traffic and Transportation Engineering (English Edition), vol. 8, no. 4, pp. 582–595, Aug. 2021, doi: 10.1016/j.jtte.2021.07.003.

[9] R. Oertel and P. Wagner, "Delay-time actuated traffic signal control for an isolated intersection," in Proc. 90th Annu. Meeting Transp. Res. Board, 2011, pp. 1–13.

[10] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, arXiv:1707.06347.

[11] C. Spatharis, and K. Blekas, "Multiagent Reinforcement Learning for Autonomous Driving in Traffic Zones with Unsignalized Intersections," Journal of Intelligent Transportation Systems, vol. 28, no. 1, pp. 103–119, Aug. 14, 2022, doi: 10.1080/15472450.2022.2109416.

[12] A. G. Vrahatis, K. Lazaros, and S. Kotsiantis, "Graph attention networks: A comprehensive review of methods and applications," *Future Internet*, vol. 16, no. 9, p. 318, 2024, doi: 10.3390/fi16090318.

[13] Y. Wang, T. Xu, X. Niu, C. Tan, E. Chen, and H. Xiong, "STMARL: A spatio-temporal multi-agent reinforcement learning approach for cooperative traffic light control," *IEEE Trans. Mobile Comput.*, vol. 21, no. 6, pp. 2228–2242, Jun. 2022.

[14] Z. Wang, G. Wu, and M. Barth, "Distributed Consensus-Based Cooperative Highway On-Ramp Merging Using V2X Communications," SAE Technical Paper 2018-01-1177, 2018.

[15] F. Ye, P. Wang, C. -Y. Chan and J. Zhang, "Meta Reinforcement Learning-Based Lane Change Strategy for Autonomous Vehicles," 2021 IEEE Intelligent Vehicles Symposium (IV), Nagoya, Japan, 2021, pp. 223-230, doi: 10.1109/IV48863.2021.9575379.

[16] S. Zhang and S. Whiteson, "DAC: The double actor-critic architecture for learning options," in Proc. Adv. Neural Inf. Process. Syst., 2019, pp. 2012–2022.