J Sustain Technol & Infra Plan- 2022
A peer-reviewed publication dedicated to advancing research and knowledge in the field of sustainable technologies and infrastructure planning.

Effective Q-Based Learning Approaches for Advanced Path Planning and Trajectory Generation in Autonomous Vehicles

Abdullah bin Hassan
Department of Engineering, Universiti Malaysia Sabah, Jalan UMS, Kota Kinabalu, Sabah, Malaysia

Ian Benitez
Electrical Engineering Department, FEU Institute of Technology

Abstract

Autonomous vehicles are a groundbreaking technology set to substantially alter transportation systems. Efficient route mapping and trajectory creation are essential for the secure and effective functioning of these vehicles. In this context, Q-based learning, specifically Q-learning, emerges as a promising approach to equip these vehicles with the ability to autonomously navigate complex and dynamic environments. This paper explores the integration of Q-learning into the domain of autonomous vehicle navigation. The methodology involves defining a state space that encapsulates critical information about the vehicle's surroundings, an action space that encompasses permissible vehicle maneuvers, and a reward function that guides the learning process by quantifying desirable outcomes and penalties. The Q-learning algorithm, which iteratively updates Q-values using the Bellman equation, enables autonomous vehicles to learn optimal policies for path planning and trajectory generation. Beyond theoretical considerations, research outcomes highlights the practical challenges associated with the real-world deployment of Q-based learning systems, emphasizing the need for continuous safety mechanisms,

simulation-based testing, and parameter tuning. Additionally, it underscores the adaptability of Q-learning to handle continuous state and action spaces through methods like deep reinforcement learning.
Keywords: Q-based Reinforcement Learnings, Autonomous Vehicles, Deep Neural Networks, Path Planning, Trajectory Identifications.

Introduction

The Intelligent Transportation Systems (ITS) integrates advanced technologies such as real-time data analytics [1], [2], communication systems and sensors. ITS aims to increase the efficiency, safety and sustainability of transportation [3]. Autonomous vehicles are a prominent component of the ITS and share a common goal of improving the reliability of vehicular networks. Kaja et al. (2021) discusses a similar approach towards network reliability of a wireless network [4].

In vehicular networks, vehicles are equipped with wireless communication modules that allow them to exchange information with each other and with infrastructure elements, such as traffic lights or road sensors [5], [6]. The primary objective is to improve situational awareness and facilitate cooperative behavior among vehicles. Given the high-speed nature of vehicular movement, ensuring a reliable and fast communication channel becomes a challenging task. Factors such as signal attenuation, interference, and high node mobility contribute to the complexities of maintaining a reliable network [7]. The network reliability for vehicular networks is discussed in (Kaja & Beard, 2020) [8]. Together, ITS and autonomous vehicles promise to redefine the future of transportation, offering safer, more convenient, and environmentally friendly travel options for individuals and communities.

The advent of autonomous vehicles has ignited a transformation in the realm of transportation, promising safer, more efficient, and sustainable mobility solutions. Central to the realization of this vision is the ability of autonomous vehicles to navigate complex and dynamic environments with precision and reliability [9]. Path planning and trajectory generation, the processes through which an autonomous vehicle determines its route and maneuvers through its surroundings, lie at the core of this capability. In this context, the integration of Q-based learning, particularly Q-learning, emerges as a promising approach to empower autonomous vehicles with the necessary intelligence to make real-time navigation decisions [10].

This paper embarks on a journey to explore the fusion of Q-based learning techniques with the domain of autonomous vehicle navigation [11]. While

traditional rule-based and deterministic approaches have been effective to some extent, they often struggle in dealing with the complexities of real-world scenarios, such as varying traffic conditions, unforeseen obstacles, and the need for adaptive decision-making. Q-learning, a reinforcement learning algorithm, offers a more adaptive and data-driven alternative by enabling autonomous vehicles to learn optimal policies for path planning and trajectory generation through interaction with their environment [12]. The methodology underlying Q-learning for autonomous vehicle navigation encompasses several key components. First and foremost, it involves defining a comprehensive state space that encapsulates critical information about the vehicle's surroundings. This state space extends beyond the vehicle's physical attributes, encompassing environmental factors, traffic conditions, and dynamic interactions with other road users. Accurate state representation is essential as it serves as the foundation upon which the vehicle's decision-making process relies [13]**.** Simultaneously, a well-defined action space is established, comprising the permissible vehicle maneuvers that can be executed. These actions include but are not limited to acceleration, steering, and braking commands. The vehicle's ability to choose appropriate actions from this space is central to its ability to navigate safely and efficiently. Moreover, the successful implementation of Q-learning for autonomous vehicle navigation necessitates the development of a reward function. The reward function acts as the guiding beacon for the learning process, providing feedback to the autonomous vehicle on the quality of its actions in a given state. Positive rewards are assigned for desirable outcomes, such as making progress towards a predefined goal or adhering to traffic rules, while negative rewards penalize actions that could lead to collisions, violations, or unsafe maneuvers. The reward function thus encapsulates the overarching objectives of safe and efficient navigation [14]**,**.

The core of Q-learning lies in its iterative learning process, wherein Q-values are updated using the Bellman equation. This equation encapsulates the principle of reinforcement learning, wherein the expected cumulative rewards associated with a particular state-action pair are updated based on the observed rewards and transitions [15]. Specifically, the Q-value for a state-action pair is updated as the weighted sum of the current Q-value, the immediate reward for the action, and the maximum Q-value achievable in the resulting state, considering a discount factor that represents the importance of future rewards. This iterative process enables the autonomous vehicle to gradually refine its policies and make more informed decisions over time. However, the application of Q-learning in the context of autonomous vehicle

navigation is not without its challenges and complexities. Safety considerations take precedence, and mechanisms must be in place to prevent dangerous situations, especially during the learning phase. The need for continuous safety mechanisms, such as collision detection and emergency braking, is paramount to ensure the well-being of passengers and other road users [16].

Furthermore, the practical deployment of Q-based learning systems in real-world environments underscores the importance of thorough simulation-based testing. Autonomous vehicles must undergo extensive testing in a simulated environment that replicates a wide range of scenarios and challenges, from congested urban streets to highway merges and adverse weather conditions. This iterative testing process is essential for validating the learned policies and fine-tuning the system's parameters to ensure robust and reliable performance in diverse conditions [17].

In addition to these considerations, it is worth highlighting the adaptability of Q-learning to handle continuous state and action spaces. Real-world autonomous vehicle navigation often involves continuous variables, such as vehicle velocity, heading angles, and distances. To address this, advanced variations of Q-learning, including deep Q-networks (DQN) and actor-critic methods, leverage deep neural networks to approximate Q-values in continuous spaces, providing a versatile solution to the challenges posed by the real world. This paper delves into the promising realm of Q-based learning for path planning and trajectory generation in autonomous vehicles. By harnessing the power of reinforcement learning, autonomous vehicles can acquire the intelligence and adaptability required to navigate the complexities of real-world environments effectively [18],. Through a meticulous exploration of state representation, action spaces, reward functions, and iterative learning processes, this research endeavors to contribute to the advancement of autonomous transportation technologies, bringing us one step closer to the realization of safe, efficient, and autonomous mobility solutions [19], [20].

Model Description of Q-Based Deep Learning for Autonomous Vehicle Navigation

In the domain of autonomous vehicle navigation, Q-based deep learning, often referred to as Deep Q-Networks (DQN), represents a powerful and adaptable approach to address the challenges of continuous state and action spaces. This model description outlines the key components and workings of a DQN-based system for path planning and trajectory generation in autonomous vehicles [21].

Model Architecture:

- Deep Neural Network (DNN): At the core of the DQN-based model is a deep neural network that approximates the Q-values associated with state-action pairs. The DNN comprises multiple layers of interconnected neurons that learn to map the continuous state space to a continuous action space while estimating the expected cumulative rewards.
- Input Layer: The input layer of the DNN receives the state representation as its input. This input includes a variety of information about the vehicle's current state and its surroundings, such as vehicle speed, position, orientation, sensor data (e.g., LiDAR, camera, radar), and environmental variables like road conditions and traffic density.
- Hidden Layers: The DNN typically consists of one or more hidden layers with a variable number of neurons. These hidden layers perform feature extraction and nonlinear transformations to capture complex relationships between the input state and the Q-values. The number of hidden layers and neurons in each layer can be customized based on the complexity of the navigation task.
- Output Layer: The output layer of the DNN produces Q-values for each action in the action space. These Q-values are continuous and represent the expected cumulative rewards for each action in the given state. In other words, the output layer provides a Q-value estimate for each possible maneuver the vehicle can execute.

Figure 1 below illustrates the model architecture of the DQN system, showcasing enhanced performance in achieving optimized tuning.

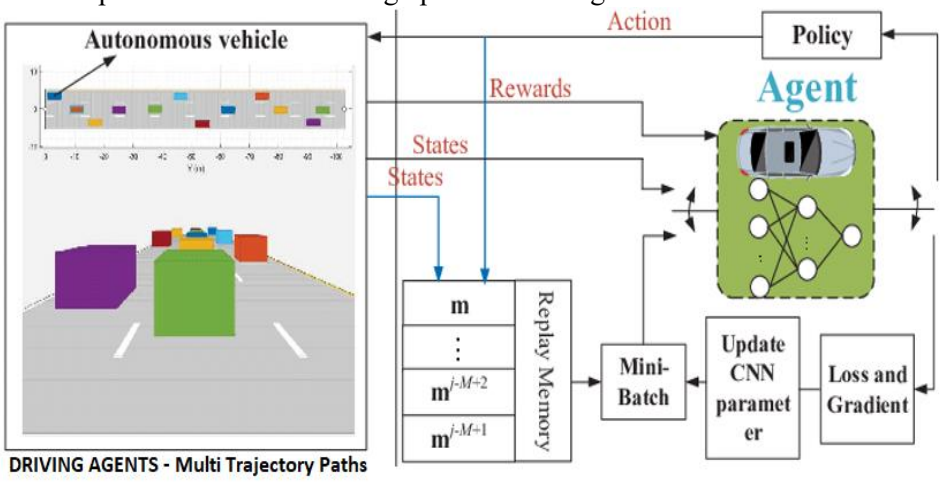

Figure 1: DQN Performance Model with fine tuning for Trajectory Planning

Model Training Process

Experience Replay: To stabilize and accelerate the learning process, the proposed DQN employs an experience replay buffer. This buffer stores past experiences, including state transitions, selected actions, rewards, and resulting states. During training, experiences are randomly sampled from the replay buffer to break the temporal correlation in the data, thereby preventing the model from overfitting to recent experiences [22]**.**

Target Network: DQN also incorporates a target network, which is a separate copy of the Q-network used for calculating target Q-values during training. The target network's parameters are periodically updated to match those of the primary Q-network. This helps stabilize the training process and prevents Q-value estimations from oscillating.

Loss Function: The loss function used to train the DQN is typically the mean squared error (MSE) between the predicted Q-values and the target Q-values. The target Q-values are calculated using the Bellman equation:

Target Q(s, a) = R(s, a) + γ * max(Q'(s', a')) ----------(1)

where Q'(s', a') represents the Q-values from the target network, s' is the next state, a' is the action chosen in the next state, R(s, a) is the immediate reward for taking action a in state s, and γ is the discount factor.

Optimization Algorithm: Common optimization algorithms like stochastic gradient descent (SGD) are employed to minimize the loss and update the weights of the DNN. The learning rate and other hyperparameters are further tuned to control the rate of convergence and stability during training.

In the context of deep neural networks for the proposed model architecture, alterations in Q-values were implemented to optimize fine-tuning, as described by Kosuru & Venkitaraman (2022) [23]. These alterations ensure operational design domains of environmental instances are influenced during the prediction of path planning. The degree of this influence is quantified by equation (2), which is presented in the Q-learning algorithm for trajectory prediction. This equation governs the shaping of the Q-factor throughout the learning process, subsequently influencing passive learning in autonomous controls. The fine-tuning, achieved through the shaping of the DQN model after state transitions, is expressed in the equation (2) as below [23].

$s_{i+1} \leftarrow I(s_i, a_i); Q(s_i, a_i) = u_{i+1} + \mu\, Q_{max}(s_{i+1}); \quad 0 \leq \mu \leq 1$ -----(2)

With reference from equation (2), the variation of Q-learning update rule for reinforcement learning on trajectory functions calculated as,

$$Q(\tau) = \Sigma \left[u_i + \mu * Q_{max(s_{i+1})}\right] for\ i = 0\ to\ T-1 \quad \text{----(3)}$$

Equation (3) sums up the immediate rewards (u_i) at each time step i and incorporates the Q-values for the next states (s_(i+1)) at each step. The parameter μ controls the trade-off between immediate rewards and expected future rewards for the entire trajectory.

In order to find the optimal trajectory, τ*, we aim to maximize the Q-value function Q(τ) by selecting actions that lead to higher expected cumulative rewards. The optimal trajectory can be found by solving cumulative rewards based on Q-values. Equation (4) represents trajectory that maximizes the expected cumulative rewards based on the learned Q-values.

$$\tau * = argmax\big(Q(\tau)\big) \quad \text{-----(4)}$$

Thus the, cumulative rewards for the entire trajectory τ by summing up the immediate rewards (u_i) at each time step i and incorporating the Q-values for the next states (s_(i+1)) at each step. The parameter μ controls the trade-off between immediate rewards and expected future rewards which is represented by equation (5) below,

$$Q(\tau) = \Sigma \left[u_i + \mu * Q_{max(s_{i+1})}\right] for\ i = 0\ to\ T-1 \quad \text{---------(5)}$$

Calculating the Learning Phase

The learning phase of Deep Q is tuned from optimization considering the learning phase where Q-values are updated through Q-learning, the evaluation of candidate trajectories based on Q-values, and the selection of the optimal trajectory. The optimal trajectory is the one that is expected to yield the highest cumulative rewards, considering both immediate rewards and the estimation of future rewards.

$$Q(\tau_{candidate}) = \Sigma \left[u_i + \mu * Q_{max(s_{i+1})}\right] for\ i = 0\ to\ T-1 \quad \text{--------(6)}$$

where T is the length of the trajectory, u_i is the immediate reward at time step i, and Q_max(s_(i+1)) is the maximum Q-value for the next state s_(i+1).

Executing the action and observe the immediate reward (u_i) and the next state (s_(i+1)). Update Q-values is calculated as follows.

$$Q(s_i, a_i) = u_i + \mu * max\big(Q(s_{i+1}, a')\big) for\ all\ a' in\ action\ space \quad \text{--------(7)}$$

Algorithm Representation for Optimal Trajectory Planning

Table 1 – Optimal Trajectory Path Planning Sequence Determination

Initialization

Initialize Q-values for state-action pairs.

Initialize an empty list to store candidate trajectories.

Learning Phase (Q-Learning)

For each episode-

Initialize the starting state.
Initialize an empty trajectory τ_candidate.
For each time step within the episode:
Select an action using the policy derived from Q-values (π(τ)):
Use argmax(Q(s_i, a)) to choose the action with the highest Q-value.
Execute the action and observe the immediate reward (u_i) and the next state (s_(i+1)).
Update Q-values using the Q-learning update rule:
Q(s_i, a_i) = u_i + μ * max(Q(s_(i+1), a')) for all a' in action space.
Add the state-action pair (s_i, a_i) to τ_candidate.
Append τ_candidate to the list of candidate trajectories.

Trajectory Evaluation

After learning, evaluate the quality of each candidate trajectory τ_candidate in the list by calculating its expected cumulative rewards based on Q-values.
Calculate Q(τ_candidate) for each τ_candidate using the formula:
Q(τ_candidate) = Σ [u_i + μ * Q_max(s_(i+1))] for i = 0 to T-1.
Here, T is the length of the trajectory, u_i is the immediate reward at time step i, and Q_max(s_(i+1)) is the maximum Q-value for the next state s_(i+1).

Optimal Trajectory Selection

Choose the trajectory with the highest expected cumulative rewards:
τ* = argmax(Q(τ_candidate)).
τ* represents the optimal trajectory.

Inference and Trajectory Generation:

Once the DQN is trained, it can be used for real-time inference and trajectory generation. Given the current state of the vehicle, the DQN predicts Q-values for all available actions. The action with the highest Q-value is selected, and the corresponding maneuver is executed [24],. This process is repeated iteratively as the vehicle navigates its environment, allowing it to make adaptive decisions in real-time while adhering to safety constraints and optimizing its path and trajectory. As the vehicle follows the planned trajectory, it executes the selected actions at each time step, navigating the environment based on the learned policies. Continuously monitor the vehicle's state and update the trajectory as necessary to adapt to changes in the environment or unexpected events [25].

In the realm of Frenet coordinates, we employ a fifth-degree polynomial as our guiding path planning method at a lower level. The starting point for this journey is

the current vehicle position, while the improved Bi-RRT path [28] helps us identify the local aim point.

Now, let's delve into the specifics. We have an initial configuration denoted as D0 with components {d0, d0·, d0··}, and a target configuration 1D1 represented by {d1, d1·, d1··}. Additionally, we factor in a designated braking time, denoted as T. With this information in hand, we can determine the quintic polynomial coefficients in the lateral direction concerning time 't'. In this context, d0, d0·, and d0·· correspond to the lateral offset, lateral velocity, and lateral acceleration, respectively.

Figure 2 below represents simulated outcome of trajectory determination for fine-tuned path planning [26].

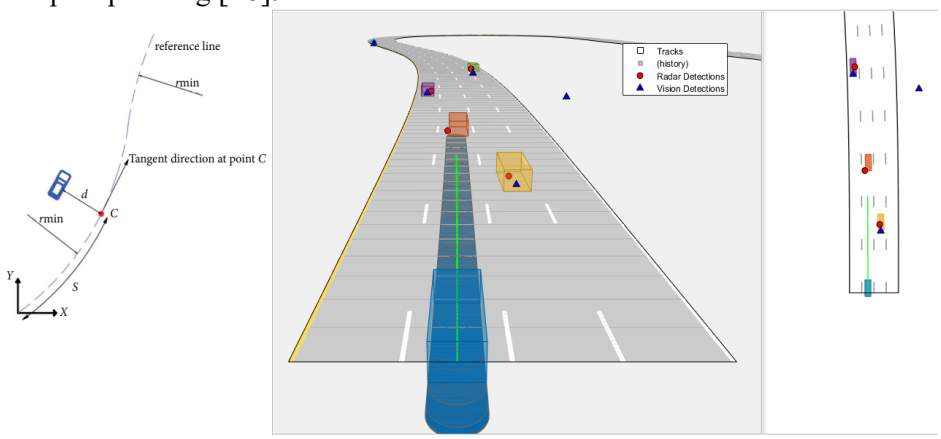

Figure 2: Simulated Trajectory Planning – AV Vehicles

Environmental Simulations

This research aims to evaluate and enhance the performance of autonomous vehicles in diverse scenarios without the need for real-world testing. A simulated frame work has established for the obtained fine tuned Q-Model for the details on experimental test procedures for research is detailed as below section [27].

Simulation Framework Setup

A comprehensive simulation environment is established, encompassing virtual representations of road networks, traffic conditions, obstacles, and environmental factors.

The autonomous vehicle's specifications, including its sensor suite, dynamic model, and control parameters, are defined within the simulation framework.

Scenario Exploration

Various navigation scenarios are meticulously designed to replicate real-world driving conditions. These scenarios range in complexity, including highway driving, urban environments, parking maneuvers, and scenarios that challenge the system's response to safety-critical situations.

Trajectory Initialization

Simulated vehicle scenarios commence with the initialization of the vehicle's initial position, orientation, and velocity within the digital environment.

An empty trajectory container, τ, is established to capture the sequence of state-action pairs that compose the planned trajectory.

Real-time Simulation Loop

The simulation follows an iterative, real-time loop, mirroring the operational pace of an autonomous vehicle.

In each simulation time step

The system continuously monitors the vehicle's state, including sensor data and dynamic parameters, within the virtual environment.

The current state information is fed into the pretrained DQN (Deep Q-Network), which generates Q-value predictions for available actions.

The action with the highest predicted Q-value is chosen as the next maneuver, emphasizing an adaptive decision-making process:

The selected action is executed in the simulation, updating the vehicle's position, orientation, and velocity.

Immediate rewards tied to the executed action and the subsequent state transition are recorded.

The trajectory τ is continuously updated with the newly formed state-action pair (s_i, a_i) for detailed analysis.

The simulation monitors termination conditions, such as goal attainment or the occurrence of safety-critical incidents.

Comprehensive Analysis

Extensive data on the trajectory τ is collected, comprising state-action sequences, executed maneuvers, observed rewards, and time-dependent vehicle states.

In-depth analysis of the trajectory data is performed to evaluate the vehicle's performance, safety, and efficiency. Critical performance metrics include trajectory smoothness, completion time, obstacle clearance distance, and adherence to traffic regulations.

The trajectory is assessed in terms of the Q-learning algorithm's adaptability to dynamic conditions and its ability to make optimal decisions.

The research identifies and studies significant scenarios or challenges encountered during the simulation experiments [28].

Scenario Variability

The research involves running simulations across various scenarios, configurations, and environmental settings to assess the robustness and adaptability of the autonomous vehicle system.

Iterative Refinement

Insights derived from simulation results guide ongoing refinement efforts in the Q-learning model, sensor data fusion techniques, control strategies, and safety mechanisms.

Continuous improvement of the system's performance and safety is achieved through an iterative cycle of simulation-based testing and analysis.

Conclusion

The advent of autonomous vehicles has ushered in a new era of transportation, one poised to reshape our urban landscapes and revolutionize the way we move from place to place. At the heart of this transformation lies the critical importance of effective path planning and trajectory generation, which are indispensable for ensuring the safe and efficient operation of these autonomous marvels [29]. In this context, the integration of Q-based learning, and specifically Q-learning, stands as a beacon of promise, offering a pathway for equipping autonomous vehicles with the ability to navigate intricate and ever-changing environments [30].

This paper has undertaken the task of delving into the intricacies of integrating Q-learning into the realm of autonomous vehicle navigation. To embark on this journey, we first establish the foundational elements of our methodology. This includes the delineation of a state space that serves as a comprehensive repository of information encapsulating the vehicle's surroundings, an action space that encompasses the repertoire of permissible vehicle maneuvers, and a reward function that serves as the guiding compass for the learning process, quantifying desirable outcomes and penalties along the way [31].

Q-learning, the keystone algorithm in our approach, operates through iterative updates of Q-values, leveraging the Bellman equation to iteratively refine the autonomous vehicle's decision-making prowess. In doing so, it learns optimal

policies for both path planning and trajectory generation. This fundamental principle represents a cornerstone in our quest for autonomous vehicle intelligence [32]. .
However, theory alone is not sufficient when addressing the multifaceted challenges associated with the real-world deployment of Q-based learning systems. This paper serves as an illuminating reminder of the practical intricacies that must be navigated on the path toward autonomous driving. Chief among these considerations is the paramount importance of continuous safety mechanisms. As autonomous vehicles operate in dynamic and unpredictable environments, the need for robust fail-safe measures remains non-negotiable. Our exploration underscores that safety and reliability must be at the forefront of any autonomous vehicle's design [33], [34].
Furthermore, the research outcomes cast a spotlight on the indispensable role of simulation-based testing. In a controlled virtual environment, autonomous vehicles can encounter a vast array of scenarios, learning from both successful interactions and near-miss situations. This invaluable data paves the way for the refinement of Q-learning algorithms, fine-tuning them for real-world deployment. It serves as a reminder that while theoretical frameworks are essential, it is within the digital realm that these concepts can be rigorously tested and validated [35].
Lastly, our investigation unveils the remarkable adaptability of Q-learning in handling continuous state and action spaces. Through the incorporation of methods like deep reinforcement learning, we witness how this versatile approach can transcend boundaries and grapple with the complex and ever-evolving landscape of autonomous vehicle navigation. This adaptability opens doors to an even brighter future, where autonomous vehicles are equipped to thrive in a diverse array of real-world scenarios [36].
In conclusion, the fusion of Q-learning and autonomous vehicle navigation represents a pivotal step in the journey toward a safer, more efficient, and more accessible mode of transportation. As we navigate the twists and turns of this transformative era, it is imperative to balance theoretical innovation with practical implementation [37].. Only by continually refining our understanding, enhancing safety measures, and embracing the versatility of Q-learning can we pave the road to a future where autonomous vehicles seamlessly navigate the complexities of our world. With unwavering commitment and a pioneering spirit, we stand on the precipice of a transportation revolution, where the autonomous vehicles of tomorrow will lead us to a brighter and more connected future.

References:

[1] L. Figueiredo, I. Jesus, and J. A. T. Machado, "Towards the development of intelligent transportation systems," *Transportation* , 2001.
[2] S. Shaheen and R. Finson, "Intelligent Transportation Systems," Dec. 2013.
[3] H. Kaja, *Survivable and Reliable Design of Cellular and Vehicular Networks for Safety Applications*. University of Missouri-Kansas City, 2021.
[4] H. Kaja, R. A. Paropkari, C. Beard, and A. Van De Liefvoort, "Survivability and disaster recovery modeling of cellular networks using matrix exponential distributions," *IEEE Trans. Netw. Serv. Manage.*, vol. 18, no. 3, pp. 2812–2824, 2021.
[5] N. Cheng, F. Lyu, J. Chen, W. Xu, H. Zhou, and S. Zhang, "Big data driven vehicular networks," *Network*, 2018.
[6] C. M. Silva, B. M. Masini, G. Ferrari, and I. Thibault, "A Survey on Infrastructure-Based Vehicular Networks," *Mobile Information Systems*, vol. 2017, Aug. 2017.
[7] R. Chen, W.-L. Jin, and A. Regan, "Broadcasting safety information in vehicular networks: issues and approaches," *IEEE Netw.*, vol. 24, no. 1, pp. 20–25, Jan. 2010.
[8] H. Kaja and C. Beard, "A Multi-Layered Reliability Approach in Vehicular Ad-Hoc Networks," *International Journal of Interdisciplinary Telecommunications and Networking (IJITN)*, vol. 12, no. 4, pp. 132–140, 2020.
[9] A. Faisal, T. Yigitcanlar, M. Kamruzzaman, and G. Currie, "Understanding autonomous vehicles: A systematic literature review on capability, impact, planning and policy," *J. Transp. Land Use*, vol. 12, no. 1, pp. 45–72, Jan. 2019.
[10] G. P. Kontoudis and K. G. Vamvoudakis, "Kinodynamic Motion Planning With Continuous-Time Q-Learning: An Online, Model-Free, and Safe Navigation Framework," *IEEE Trans Neural Netw Learn Syst*, vol. 30, no. 12, pp. 3803–3817, Dec. 2019.
[11] S. Dhakal, D. Qu, D. Carrillo, and Q. Yang, "Oasd: An open approach to self-driving vehicle," *on Connected and …*, 2021.
[12] M. J. Van Nieuwstadt and R. M. Murray, "Real-time trajectory generation for differentially flat systems," *Int. J. Robust Nonlinear Control*, vol. 8, no. 11, pp. 995–1020, Sep. 1998.
[13] W. Weng, H. Gupta, N. He, L. Ying, and R. Srikant, "The mean-squared error of double Q-learning," *Adv. Neural Inf. Process. Syst.*, vol. 33, pp. 6815–6826, 2020.
[14] D. Malyuta *et al.*, "Convex Optimization for Trajectory Generation," *arXiv [math.OC]*, 16-Jun-2021.

[15] W. Huang, K. Wang, Y. Lv, and F. Zhu, "Autonomous vehicles testing methods review," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, 2016, pp. 163–168.
[16] M. B. Milam, K. Mushambi, and R. M. Murray, "A new computational approach to real-time trajectory generation for constrained mechanical systems," in *Proceedings of the 39th IEEE Conference on Decision and Control (Cat. No.00CH37187)*, 2000, vol. 1, pp. 845–851 vol.1.
[17] D. Ding, X. Fan, Y. Zhao, K. Kang, Q. Yin, and J. Zeng, "Q-learning based dynamic task scheduling for energy-efficient cloud computing," *Future Gener. Comput. Syst.*, vol. 108, pp. 361–371, Jul. 2020.
[18] F. Duarte and C. Ratti, "The Impact of Autonomous Vehicles on Cities: A Review," *Journal of Urban Technology*, vol. 25, no. 4, pp. 3–18, Oct. 2018.
[19] J. Hu and M. P. Wellman, "Nash Q-learning for general-sum stochastic games," 2003. [Online]. Available: https://www.jmlr.org/papers/volume4/temp/hu03a.pdf. [Accessed: 14-Sep-2023].
[20] D. S. Leslie and E. J. Collins, "Individual Q-Learning in Normal Form Games," *SIAM J. Control Optim.*, vol. 44, no. 2, pp. 495–514, Jan. 2005.
[21] S. Kato, E. Takeuchi, Y. Ishiguro, Y. Ninomiya, K. Takeda, and T. Hamada, "An Open Approach to Autonomous Vehicles," *IEEE Micro*, vol. 35, no. 6, pp. 60–68, Nov. 2015.
[22] L. De Filippis, "Advanced path planning and collision avoidance algorithms for UAVs," 2012.
[23] V. S. R. Kosuru and A. K. Venkitaraman, "Developing a Deep Q-Learning and Neural Network Framework for Trajectory Planning," *EJENG*, vol. 7, no. 6, pp. 148–157, Dec. 2022.
[24] M. W. Mueller, M. Hehn, and R. D'Andrea, "A Computationally Efficient Motion Primitive for Quadrocopter Trajectory Generation," *IEEE Trans. Rob.*, vol. 31, no. 6, pp. 1294–1310, Dec. 2015.
[25] R. Róka, *Advanced Path Planning for Mobile Entities*. BoD – Books on Demand, 2018.
[26] M. Hehn and R. D'Andrea, "Real-Time Trajectory Generation for Quadrocopters," *IEEE Trans. Rob.*, vol. 31, no. 4, pp. 877–892, Aug. 2015.
[27] D. J. Fagnant and K. Kockelman, "Preparing a nation for autonomous vehicles: opportunities, barriers and policy recommendations," *Transp. Res. Part A: Policy Pract.*, vol. 77, pp. 167–181, Jul. 2015.
[28] J. Wang, L. Zhang, Y. Huang, and J. Zhao, "Safety of Autonomous Vehicles," *Journal of Advanced Transportation*, vol. 2020, Oct. 2020.

[29] S. X. Yang and M. Meng, "Neural network approaches to dynamic collision-free trajectory generation," *IEEE Trans. Syst. Man Cybern. B Cybern.*, vol. 31, no. 3, pp. 302–318, 2001.
[30] Z. Li, Y. Lu, Y. Shi, Z. Wang, W. Qiao, and Y. Liu, "A Dyna-Q-Based Solution for UAV Networks Against Smart Jamming Attacks," *Symmetry* , vol. 11, no. 5, p. 617, May 2019.
[31] D. Spensieri, J. S. Carlson, R. Bohlin, and R. So¨derberg, "Integrating Assembly Design, Sequence Optimization, and Advanced Path Planning," *ASME 2008 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pp. 73–81, Jul. 2009.
[32] J. Peeters, B. Bogaerts, S. Sels, B. Ribbens, J. J. J. Dirckx, and G. Steenackers, "Optimized robotic setup for automated active thermography using advanced path planning and visibility study," *Appl. Opt.*, vol. 57, no. 18, pp. D123–D129, Jun. 2018.
[33] H. H. Viet, S. H. An, and T. C. Chung, "Dyna-Q-based vector direction for path planning problem of autonomous mobile robots in unknown environments," *Adv. Robot.*, vol. 27, no. 3, pp. 159–173, Feb. 2013.
[34] M. Werling, J. Ziegler, S. Kammel, and S. Thrun, "Optimal trajectory generation for dynamic street scenarios in a frenet frame," in *2010 IEEE international conference on robotics and automation*, 2010, pp. 987–993.
[35] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 2520–2525.
[36] M. Hehn and R. D'Andrea, "Quadrocopter Trajectory Generation and Control," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 1485–1491, Jan. 2011.
[37] T. Kröger, "Literature Survey: Trajectory Generation in and Control of Robotic Systems," in *On-Line Trajectory Generation in Robotic Systems: Basic Concepts for Instantaneous Reactions to Unforeseen (Sensor) Events*, T. Kröger, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 11–31.