Deep Reinforcement Learning for Autonomous Driving Decision-Making in Webots

1st Xin Xing
Faculty of Electrical Engineering and Information Technology
Ostfalia University of Applied Sciences
Wolfenbuettel, Germany
xi.xing@ostfalia.de

2nd Sebastian Ohl
Faculty of Electrical Engineering and Information Technology
Ostfalia University of Applied Sciences
Wolfenbuettel, Germany
s.ohl@ostfalia.de

***Abstract*—Reinforcement learning (RL) systems for autonomous driving require carefully designed simulation environments to facilitate training and testing in dynamic and complex scenarios. This paper presents the architecture of a simulation environment tailored to train autonomous vehicle decision-making systems, particularly for maneuver-based decisions such as Adaptive Cruise Control (ACC) and Automatic Emergency Braking (AEB). We emphasize the integration of key components, including vehicle sensors, controllers, and environment models, with a specific focus on the real-time communication required to support RL training. Utilizing the Webots simulator and the ExerShuttle project as a case study, this paper demonstrates how RL-based agents can be integrated into a simulated driving environment, addressing challenges in data flow accuracy, system responsiveness, and component interoperability. The proposed architecture provides a scalable and flexible framework for developing robust RL models that adapt to diverse driving conditions, supporting the advancement of safe and efficient autonomous systems.**

***Index Terms*—Simulation Environment, Autonomous Driving, Reinforcement Learning, Webots Simulator, Communication Architecture.**

I. INTRODUCTION

The advancement of autonomous driving systems necessitates the development of comprehensive simulation environments to assess, verify, and refine decision-making algorithms before their deployment in real-world scenarios. The effectiveness of these systems depends not only on decision-making models, such as Reinforcement Learning (RL), but also on the architectural design of the environment in which they are trained and tested. Creating robust and realistic simulation environments is essential for understanding how these systems perform in dynamic and complex settings, where real-time decision-making is critical for both safety and efficiency [1] [2].

A key challenge in autonomous system development lies in constructing an integrated environment that accurately replicates real-world conditions. This includes simulating sensors, communication protocols, vehicle dynamics, and interactions with dynamic entities such as pedestrians and other vehicles. Several simulation platforms, including Webots, CARLA, Autoware, and Apollo, have been developed to support autonomous driving research. While CARLA and Apollo offer high-fidelity urban simulations with detailed traffic and environmental dynamics, Webots provides flexibility and ease of use, making it well-suited for smaller-scale, multi-agent systems and early-stage prototyping [3].

Although simulation platforms play a crucial role, prior research has mainly focused on decision-making algorithms and sensor fusion techniques, paying comparatively less attention to the supporting system architecture. This includes the structure of the simulation environment and the communication mechanisms between system components. This paper seeks to address this gap by examining the architectural requirements for developing effective simulation environments for autonomous vehicles, emphasizing component communication, sensor integration, and system scalability.

II. RELATED WORK

Research on autonomous driving has focused on decision-making algorithms and their supporting architectures. Early studies used rule-based approaches like car-following models [4], which were effective in simple scenarios but lacked adaptability.

With AI advancements, Deep Reinforcement Learning (DRL) has emerged for handling complex driving environments. Sallab [5] highlights DRL's potential, emphasizing the need for well-designed simulation environments. Platforms like CARLA, Apollo, and Autoware support RL-based testing in urban environments. CARLA [6] provides high-fidelity urban simulations, while Autoware [7] and Apollo [8] offer modular frameworks for autonomous vehicle development.

For large-scale traffic simulations, SUMO [9] efficiently models vehicle interactions. Webots [10], in contrast, is a flexible, user-friendly platform for multi-agent simulations and early-stage prototyping. Unlike CARLA, which specializes in high-fidelity urban driving, Webots supports diverse robotic systems and RL frameworks like OpenAI Gym.

Beyond simulation platforms, research emphasizes system architecture and component communication. OpenAI Gym [11] highlights the importance of seamless agent-environment interaction, crucial for Webots's multi-agent and sensor management. Comparative studies [12] indicate that while Webots excels in flexibility, CARLA and Apollo are better suited for large-scale urban driving simulations.

Webots, CARLA, Autoware, and Apollo are key simulation platforms, each with distinct advantages. Webots excels in rapid prototyping with an integrated development environment, supporting diverse robotic platforms and sensors. Its lightweight, flexible design and lower computational demands make it ideal for iterative development and academic research.

In contrast, CARLA, Autoware, and Apollo focus on high-fidelity simulations, replicating real-world conditions to test edge cases and system robustness. While offering unparalleled realism, their complexity and high computational demands can be challenging for early-stage prototyping. Webots, with its lightweight architecture, ease of use, and broad applicability, remains a highly efficient and accessible simulation environment for researchers and developers.

While decision-making algorithms have advanced, the simulation architectures supporting them remain a critical research area. Simulation tools are essential for testing and validating autonomous systems, providing a safe, cost-effective environment for refining algorithms before deployment.

III. Background

In our preceding study [13], we validated the decision-making system using the Policy Gradient (PG) and Proximal Policy Optimization (PPO) algorithms, thereby demonstrating their efficacy in autonomous decision-making. This paper builds upon our previous work by incorporating the Deep Deterministic Policy Gradient (DDPG) algorithm to investigate the viability of RL training within the Webots environment. This provides a more comprehensive assessment of the applicability of RL in autonomous systems.

Deep Deterministic Policy Gradient is an actor-critic algorithm that has been developed for use in environments with continuous action spaces. It has the ability to bridge the gap between policy-based and value-based RL. DDPG employs deep neural networks to represent both the policy (actor) and the value (critic) functions, thereby enhancing the algorithm's capacity to process high-dimensional sensory inputs, which are prevalent in applications such as robotics and autonomous vehicles. The following formulas have been derived from [14].

In the context of the DDPG framework, the critic network updates its parameters by minimising the mean squared error between the predicted Q-values and the target Q-values, which are computed using the Bellman equation [14]. The loss function is expressed as follows:

$$L = \frac{1}{N}\sum_{i}\left(y_i - Q(s_i, a_i|\theta^Q)\right)^2, \tag{1}$$

where $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$, encapsulating the Bellman equation for continuous actions with target networks Q' and μ' to mitigate oscillations in learning.

The policy network, or the actor, is updated by employing the policy gradient method. The gradient of the policy's performance is estimated by the product of the gradients of the critic's Q-value with respect to the action, evaluated at the current policy's action, and the gradient of the actor network with respect to its parameters. This relationship is quantified as

$$\nabla_{\theta^\mu} J \approx \frac{1}{N}\sum_{i} \nabla_a Q(s, a|\theta^Q)|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^\mu}\mu(s|\theta^\mu)|_{s_i}, \tag{2}$$

effectively linking the actor's policy updates directly to the performance improvements as assessed by the critic.

Furthermore, DDPG stabilizes the learning process by softly updating the target networks, which are critical to mitigate the effects of rapidly changing policy estimates that could lead to destructive interference. The parameters of the target networks for both the actor and the critic are updated using the formula

$$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}, \quad \theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}, \tag{3}$$

using a soft update method to slowly integrate the learned networks' weights into the target networks, enhancing the stability of the training process.

The integration of a replay buffer to randomly sample previous experiences and the employment of target networks for soft updates are innovations that allow DDPG to outperform other algorithms in complex environments. This approach allows the agent to learn stable policies from raw sensory inputs without requiring manual feature engineering.

IV. Methodology

The Webots simulator comprises a number of key elements, including the capacity for dynamic environmental interaction, sensors for data acquisition, and modules for localisation, map integration, perception, and planning, as shown in Figure 1. At the core of the system is an RL agent that processes perceptual data in order to make decisions, which are then executed by the control and actuation system to manage the vehicle's steering and velocity, thereby ensuring adaptive and efficient navigation.

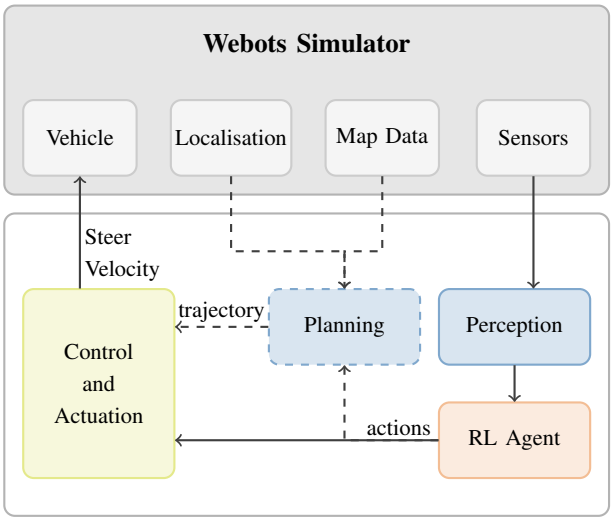

Fig. 1. Architecture for reinforcement learning in Webots simulator

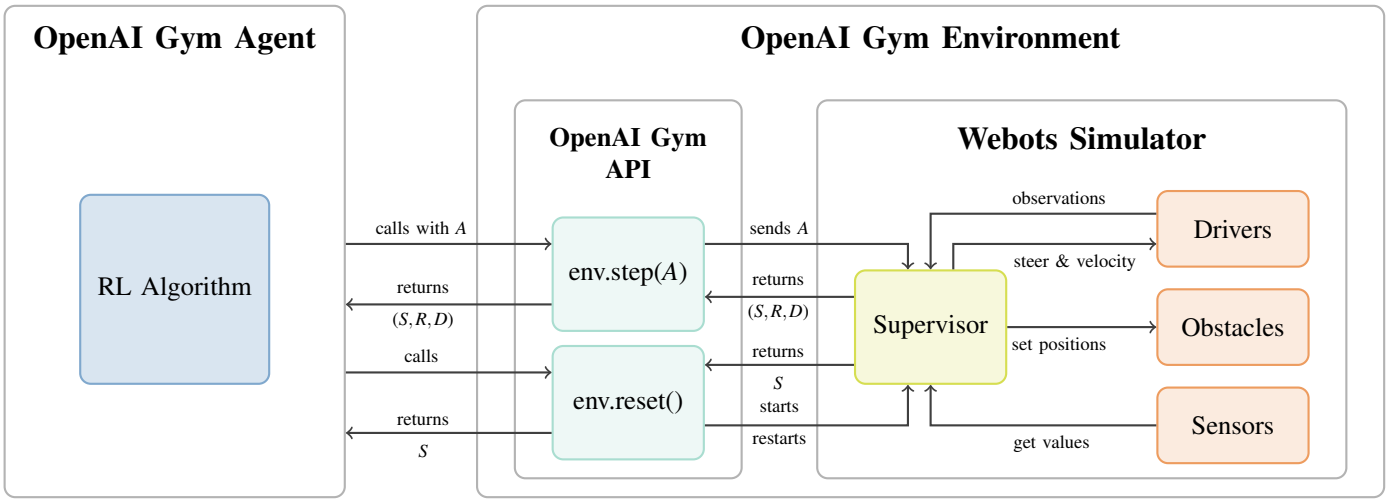

Fig. 2. Integrating Deep Deterministic Policy Gradient with OpenAI Gym and Webots for Autonomous Driving Simulation

A. Environment Modeling

Webots provides powerful tools for creating realistic simulation environments. In autonomous driving projects like ExerShuttle, which operates in urban settings, Webots enables the accurate replication of test-site layouts, including roads, intersections, pedestrian areas, and obstacles. Users can model different terrains, buildings, and traffic elements to ensure a close match to real-world driving conditions. Additionally, the simulation can be adjusted for various weather and lighting conditions, allowing performance evaluation in scenarios such as low-light or rainy environments.

B. Sensor and Actuator Integration

Webots supports a wide range of sensors and actuators essential for autonomous driving, including cameras, LiDARs, Radars, GPS, and IMUs. These sensors collect environmental data, enabling real-time decision-making by the RL agent. They can be attached to virtual vehicle models to replicate real-world sensor placements.

In the ExerShuttle project, LiDAR is used for object detection and distance measurement, generating point cloud data to identify surrounding objects. A target selection mechanism identifies and prioritizes obstacles in the shuttle's path, such as other vehicles, based on proximity and motion prediction. Cameras assist in recognizing road markings and signs, improving lane-keeping and traffic compliance. The data from these sensors feed into the RL model, allowing the vehicle to navigate, avoid obstacles, and adjust speed in a realistic manner.

C. Communication and Control Systems

A key strength of Webots is its ability to simulate complex control systems and facilitate communication between different vehicle components. The platform supports custom controllers written in multiple languages, including C, C++, Python, and MATLAB, allowing for flexible and project-specific control logic implementation. This modular approach is particularly beneficial for projects like ExerShuttle, where separate controllers can manage different functions, such as Adaptive Cruise Control (ACC) [15, p. 24] and Automatic Emergency Braking (AEB) [15, p. 666].

In a RL setup, Webots enables control systems to receive sensor data, process it, and send commands to the vehicle's actuators. For instance, if LiDAR detects an obstacle, the system can trigger the emergency braking controller, bringing the vehicle to a stop. This setup closely mirrors real-world autonomous vehicle control mechanisms.

D. Constructing an OpenAI Gym Environment Using Webots

The training architecture integrating the RL algorithm within an OpenAI Gym and Webots simulation environment is shown in Figure 2, specifically tailored for autonomous driving applications.

1) Agent and Environment Interface: At the core of this architecture is the OpenAI Gym Agent, which employs an RL algorithm to optimize driving decisions based on state feedback and reward mechanisms. The agent interacts with the environment through the OpenAI Gym API, which provides two key functions:

- env.step(A): Advances the simulation by one timestep using action A, determined by the policy. This function returns the new state S, reward R, and a done flag D, which indicates whether the episode has ended.
- env.reset(): Reinitializes the environment, starting a new learning episode from a default state.

2) Simulator Dynamics: Webots provides a realistic 3D simulation environment for various driving scenarios, integrating several essential components:

- Supervisor: Manages the simulation by controlling parameters, monitoring vehicle behavior, and overseeing interactions with environmental features.
- Drivers: Execute the driving commands issued by the agent, influencing the vehicle's steering and velocity.

- Obstacles: Represent static and dynamic objects, adding realistic driving challenges.
- Sensors: Capture environmental data like distance to obstacles, traffic signals, or road markings, feeding this information back to the supervisor to inform the state representation.

3) Feedback Loop: The feedback loop plays a crucial role in RL-based learning. The agent's actions influence the vehicle's behavior through the driver module, while sensors capture the resulting interactions. The supervisor processes this data, generating a new state, reward, and completion status, which are relayed to the agent via the OpenAI Gym API. This iterative process enables continuous policy refinement.

4) Learning and Adaptation: Through repeated interaction with the environment, the agent continuously refines its policy based on accumulated experiences. This architectural design allows for extensive experimentation with diverse driving strategies, enhancing and validating autonomous driving algorithms in a controlled yet realistic setting.

E. Real-Time and Multi-Agent Simulation

Webots supports real-time and multi-agent simulations, allowing multiple autonomous entities—such as vehicles, pedestrians, and cyclists—to interact within a shared environment. In the ExerShuttle project, this enables realistic testing of vehicle responses in dynamic, multi-agent scenarios.

Real-time simulation is particularly valuable for RL, as it presents agents with evolving conditions and unpredictable interactions. This capability closely mirrors real-world autonomous driving environments, where vehicles must react to multiple stimuli simultaneously, ensuring robust and adaptable decision-making.

V. Evaluation

The following evaluation aims to assess the feasibility and effectiveness of utilising the Webots simulation platform for the training of RL models in autonomous driving tasks. The principal objective of this evaluation is to ascertain the suitability of Webots as a training environment. This encompasses an assessment of its capacity to simulate intricate driving scenarios, integrate with RL frameworks, and accommodate multi-sensor configurations that are pivotal for decision-making models.

A. Training the Algorithm

The training of an autonomous driving system using RL focuses on developing behaviors such as ACC and AEB. The system dynamically interacts with its environment, processing sensory inputs, making decisions based on learned policies, and optimizing actions through a reward mechanism.

The architecture consists of:

- Perception Layer: Aggregates sensor data (e.g., vehicle speed, traffic conditions) and extracts key features for decision-making.
- Decision-Making Layer: Utilizes an RL framework to map states to actions, optimizing long-term rewards. ACC learns optimal following distances and acceleration patterns, while AEB prioritizes rapid deceleration in high-risk scenarios, overriding ACC when necessary.
- Feedback Loop: Continuously refines policies based on interactions, ensuring adaptability to varied driving conditions.

This iterative training process enhances system robustness, enabling dynamic prioritization of actions. As the model matures, it generalizes beyond initial behaviors, integrating new driving strategies to improve safety and adaptability.

The following Figures 3 and 4 illustrate specific training cases within the Webots environment:

- ACC Training Scenario: Figure 3 demonstrates the training process for ACC, where the Autonomous Vehicle (AV), using the Intelligent Driver Model (IDM) [15, p. 148] for speed control, learns to adjust acceleration and maintain a safe distance from the Leading Vehicle (LV) in a dynamic traffic environment.
- AEB Training Scenario: Figure 4 displays the AEB training case. Here, the AV is trained to recognize and respond to potential collision scenarios by applying maximum negative acceleration in emergency situations, thus, enhancing vehicle safety.

The aforementioned scenarios serve to illustrate Webots's aptitude for facilitating a multitude of RL tasks, which are of essential importance to the advancement of autonomous driving technologies.

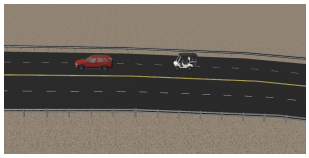

Fig. 3. AV follows the LV

Fig. 4. AV brakes emergency

B. Evaluation Metrics

To evaluate Webots as an RL training environment, we focused on the following metrics:

- Simulation Fidelity: The realism of the driving environment, including road conditions, obstacles, and dynamic elements like pedestrians and other vehicles.
- Sensor Integration: The accuracy and responsiveness of sensor data and its usability within RL training workflows.
- Training Stability: The consistency and stability of RL model convergence within Webots, including the number of episodes required for convergence.
- Environment Responsiveness: The real-time feedback loop efficiency between the RL agent and the simulated environment, especially under high interaction scenarios.

C. Simulation Fidelity and Realism

Webots enables the development of highly customizable driving environments. The ExerShuttle test area at Am Exer is simulated using OpenStreetMap data, continuously refined

to reflect real-world updates, such as lane modifications, new constructions, and parking areas. This allows for an assessment of Webots' capability in replicating real-world driving complexities. In early RL training phases, only essential elements (e.g., lanes, autonomous vehicles, key traffic participants) are included.

The ExerShuttle vehicle is also simulated using Webots's Ackermann vehicle model, with key parameters (e.g., weight, center of gravity, steering angle, speed, acceleration) calibrated to match the real vehicle. The integration of its CAD model further enhances fidelity, ensuring a realistic simulation aligned with actual autonomous vehicle dynamics.

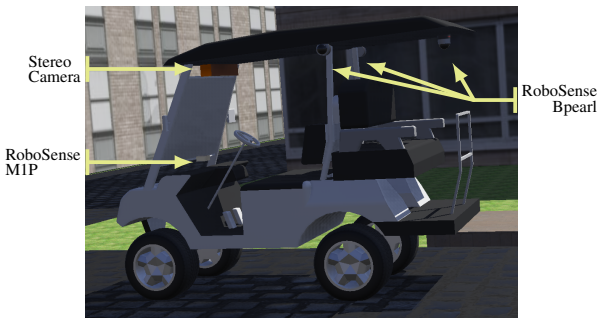

Fig. 5. Autonomous Vehicle and Sensor Integration

D. Sensor Integration

In the ExerShuttle project, a suite of sensors has been optimized to ensure comprehensive environmental perception as shown in Figure 5. The RoboSense M1P solid-state LiDAR, with a range of 150 m, monitors the front, enabling early detection of obstacles and road features. Three RoboSense Bpearl blind-spot LiDARs, each covering 30 m, enhance side and rear awareness, minimizing blind spots. Additionally, the Allied Vision ScanScene Pro stereo camera captures details within a 10 m radius, improving navigation in dense environments. Together, these sensors provide a 360° perception for safer and more efficient autonomous driving.

To ensure accurate LiDAR simulation, custom PROTO files were created to model the M1P and Bpearl sensors with precise specifications, including Field of View (FOV), number of channels, and range. Since direct stereo camera support is unavailable, a stereo vision setup is emulated using two RGB cameras with a RangeFinder to approximate depth information, generating realistic 3D scene data.

Integrated GPS and IMU modules in the simulation provide precise vehicle pose data, eliminating real-world localization errors. As a result, the ExerShuttle's Novatel GNSS/INS module was not replicated, as Webots's built-in sensors ensure accurate positioning, maintaining realism while optimizing simulation efficiency.

E. Training Stability and Convergence Analysis

The stability of RL models is crucial for their successful deployment, particularly in simulation environments like Webots. Stability refers to the model's ability to consistently converge across training sessions, assessed through the analysis of reward trends over time. Reliable convergence is essential for ensuring that an RL model can achieve an optimal or near-optimal policy within a predefined number of training episodes.

In Webots, the required number of episodes for convergence directly influences training efficiency and the practical feasibility of real-world deployment. A stable training process enhances model reliability, reducing variability in learning outcomes and improving the predictability of autonomous decision-making. The DDPG agent comprises actor and critic networks with two hidden layers of 256 units (initialized uniformly in $\pm 3 \times 10^{-3}$), employs learning rates of 10^{-4} for the actor and 10^{-3} for the critic, uses a replay buffer of 6×10^4 samples and batch size 64, injects Ornstein–Uhlenbeck noise $(\theta = 0.15, \mu = 0, \sigma = 0.2)$ with ε-greedy exploration (linearly decaying ε from 200), performs soft target updates with $\tau = 10^{-3}$, and discounts rewards by $\gamma = 0.99$.

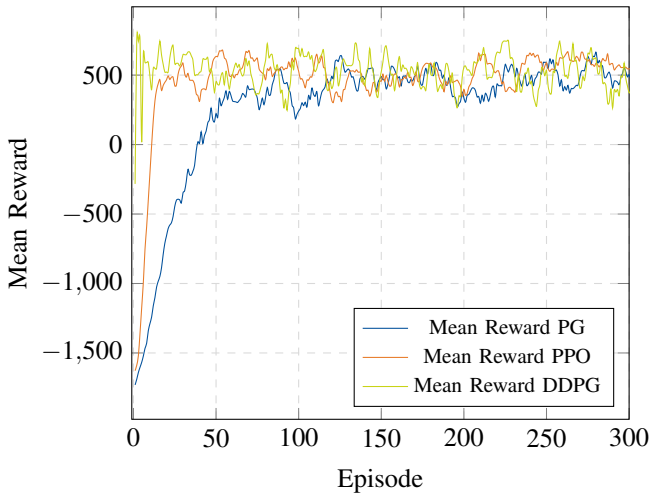

Fig. 6. Training results of PG, PPO and DDPG models

Figure 6 presents the reward trends of PG, PPO, and DDPG over the initial 300 episodes from 1000 episodes, capturing the critical convergence phase. Analyzing early-stage performance provides insights into stability and training efficiency within the Webots RL environment:

- Policy Gradient: Initially exhibits moderate reward variability before stabilizing. While capable of convergence, PG requires more episodes for stability, suggesting the need for hyperparameter tuning to improve training efficiency.
- Proximal Policy Optimization: Shows steady reward improvement and faster convergence than PG, demonstrating stability and adaptability within Webots. Its consistent learning trajectory highlights Webots's effectiveness in facilitating stable RL training.
- Deep Deterministic Policy Gradient: Experiences high initial fluctuations but converges rapidly. This reflects Webots's suitability for continuous control tasks, as DDPG efficiently processes multi-sensor data and interaction feedback, achieving stable learning in fewer episodes.

These reward trends collectively emphasize Webots's capabilities as an RL training platform. Its support for stable learning across varied RL algorithms highlights its adaptability and robustness for complex, autonomous driving tasks, affirming its practical relevance in real-world model deployment.

F. Environment Responsiveness and Feedback Loop Efficiency

Achieving training stability and convergence is crucial for developing robust RL models applicable to real-world scenarios. A key factor influencing training efficiency is the triggering frequency, which determines how often the simulation updates. While Webots theoretically supports 50 Hz, practical CPU limitations can significantly reduce this rate, leading to delayed feedback and slower convergence. This highlights the sensitivity of RL performance to simulation fidelity.

Simulation complexity also impacts computational demands, requiring a balance between environmental detail and training feasibility. Higher model complexity, characterized by increased parameters and state space dimensionality, prolongs convergence due to greater computational requirements. Advanced hardware, such as high-performance GPUs, can mitigate these challenges by improving processing efficiency and reducing training time.

In practical tests, a 50 Hz update rate was set, aiming for 20 ms per step. To maintain realistic vehicle dynamics, control steps were adjusted to 100 ms. However, processing delays of 120 ms per update reduced the effective frequency to 8.33 Hz, significantly below the target, affecting training efficiency and model responsiveness.

G. Feasibility Assessment and Limitations

Our evaluation demonstrates that Webots is well-suited for training RL-based autonomous driving models, offering a flexible and customizable simulation environment for early-stage development. However, limitations were observed, including constraints in large-scale urban simulations and potential latency in high-interaction scenarios. Future improvements should focus on enhancing simulation fidelity and responsiveness to better support complex RL tasks.

VI. Conclusions and Future Works

This paper presents a proof of concept demonstrating the feasibility and effectiveness of RL models within the Webots simulation platform for autonomous driving tasks, with a particular focus on ACC and AEB functionalities. The results confirm that Webots provides a scalable and robust environment for RL model training, enabling the system to respond adaptively and make critical decisions in diverse driving scenarios. Nevertheless, the current configuration is constrained by the simplicity of traffic dynamics and the limited complexity of environmental interactions.

In future work, the intention is to integrate SUMO in order to model more complex traffic environments, thereby enabling the simulation of a greater variety of decision-making scenarios, including advanced manoeuvres such as overtaking and intersection. This will further enhance the training scenarios and improve the system's capacity to handle real-world complexities. Furthermore, future research will investigate methodologies for transferring the RL models trained in the simulation environment to the physical ExerShuttle platform, addressing the practical challenges of real-world application and testing. In the next six months, the initial focus will be on deploying and testing simplified ACC and AEB decision models on the physical vehicle system. This phase will involve integrating the models, conducting real-world tests, and optimizing their performance to ensure reliability and adaptability under real driving conditions.

References

[1] A. Kendall, J. Hawke, D. Janz, P. Mazur, D. Reda, J.-M. Allen, V.-D. Lam, A. Bewley, and A. Shah, "Learning to Drive in a Day," 2019 International Conference on Robotics and Automation (ICRA), canada, 2018, pp. 8248–8254.

[2] R. Gutiérrez-Moreno, R. Barea, E. López-Guillén, J. Araluce, and L. M. Bergasa, "Reinforcement Learning-Based Autonomous Driving at Intersections in CARLA Simulator," *Sensors*, vol. 22, no. 21, 2022, p. 8373.

[3] R. Underwood, Q. -H. Luu and H. Liu, "A Metamorphic Testing Framework and Toolkit for Modular Automated Driving Systems," 2023 IEEE/ACM 8th International Workshop on Metamorphic Testing (MET), Melbourne, Australia, 2023, pp. 17–24.

[4] M. Brackstone and M. McDonald, "Car-following: a historical review," *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 2, no. 4, 1999, pp. 181–196.

[5] A. Sallab, M. Abdou, E. Perot, and S. Yogamani, "Deep Reinforcement Learning framework for Autonomous Driving," Electronic Imaging, vol. 2017, no. 70-76, 2017, pp. 70–76.

[6] A. Dosovitskiy, G. Ros, F. Codevilla, A. M. López, and V. Koltun, "CARLA: An Open Urban Driving Simulator," 1st Conference on Robot Learning (CoRL 2017), Mountain View, United States, 2017, pp. 1–16.

[7] S. Kato, S. Tokunaga, Y. Maruyama, S. Maeda, M. Hirabayashi, Y. Kitsukawa, A. Monrroy, T. Ando, Y. Fujii, and T. Azumi, "Autoware on Board: Enabling Autonomous Vehicles with Embedded Systems," 2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS), 2018, pp. 287–296.

[8] Z. Peng, J. Yang, T.-H. Chen, and L. Ma, "A first look at the integration of machine learning models in complex autonomous driving systems: a case study on Apollo," 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE 2020), New York, NY, USA, 2020, pp. 1240—1250.

[9] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker-Walz, "Recent Development and Applications of SUMO - Simulation of Urban MObility," International Journal On Advances in Systems and Measurements, vol. 5, no. 3&4, 2012, pp. 128–138.

[10] M. Olivier, "Cyberbotics Ltd - Webots™: Professional Mobile Robot Simulation," *International Journal of Advanced Robotic Systems*, vol. 1, no. 1, 2004, pp. 40–43.

[11] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "OpenAI Gym," arXiv preprint arXiv:1606.01540, 2016.

[12] Z. Zhong, Y. Tang, Y. Zhou, V. de Oliveira Neves, Y. Liu, and B. Ray, "A Survey on Scenario-Based Testing for Automated Driving Systems in High-Fidelity Simulation," arXiv preprint arXiv:2112.00964, 2021.

[13] X. Xing and S. Ohl, "Application of a Maneuver-Based Decision Making Approach for an Autonomous System Using a Learning Approach," ADVCOMP 2024, The Eighteenth International Conference on Advanced Engineering Computing and Applications in Sciences, Venice, Italy, Sept. 29 - Oct. 3, 2024, pp. 11–16.

[14] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," arXiv preprint arXiv:1509.02971, 2019.

[15] A. Eskandarian, "Handbook of Intelligent Vehicles," 1st ed., Springer London, 2012. ISBN: 978-0-85729-084-7 (Hardcover), 978-0-85729-085-4 (eBook).