Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 45 No. 1 (2024)

Development of F1 Tenth-specific Autonomous Navigation Algorithm

[1a]Shripad Bhatlawande, [1b]Harshal Daigavhane, [1c]Kuldeep Aher, [1d]Swarali Damle, [1e]Swati Shilaskar

[1]*Department of Electronics and Telecommunication Engineering*
Vishwakarma Institute of Technology
Pune, 411037, Maharashtra, India.

Abstract: **-** This research proposes an autonomous navigation algorithm for the ROS-based F1 Tenth racing platform, designed to excel in dynamic and challenging racing environments. It utilizes LiDAR data downsampling and a custom Gym environment to enable reinforcement learning-based control. The novelty in this research is the development of an efficient LiDAR data downsampling algorithm. Unlike traditional methods, this novel approach ensures computational efficiency, retaining essential information and preserving the graph's geometric shape. Comparative computation analysis with normal sampling and polyline approximation demonstrates that the proposed approach is 217 times faster than polyline approximation and is efficient in preserving the valley and peak points despite a slightly higher RMSE. This algorithm is effective as downscaled data also needs to be of a fixed size for RL model training. The algorithm easily integrates with the F1/10 simulator through ROS2, enabling rapid algorithm development and testing in a simulated environment. This integration speeds up development and enables thorough testing before real-world deployment. After training for 700 thousand timesteps, the best trained model was able to successfully navigate 80% of the track length. The project contributes to advancing autonomous technology and its application in dynamic, fast-paced environments, where quick decision-making and precise navigation are crucial.

Keywords*: F1/10, autonomous navigation, LiDAR down-scaling.*

1. Introduction

Autonomous cars, a revolutionary advancement in automotive technology, have transformed the way we perceive transportation. The expected global valuation of self-driving vehicles is estimated to reach $615 billion till 2026 [1]. Beyond everyday applications, autonomous racing has emerged as a thrilling extension of this technology. Automated racing aims to challenge autonomous car boundaries by demonstrating the power of artificial intelligence in the field of racing. It holds the promise of transforming the motorsports landscape and delivering positive impacts to the economy and society [2]. In these races, advanced autonomous software replaces human drivers, drawing upon the perception, planning and control pipeline which is a vital framework that involves but not limited to sensor data interpretation, planning dynamic trajectories, decision-making and adjusting performance for high-speed racing on the track, including self-assessment based on tire wear, temperature, and weather [3], [4]. Among these competitions, one notable example is F1/10, which stands as a distinctive form of autonomous racing dedicated to small-scale, 1/10th-scale self-driving cars. The F1/10 simulator has been developed with a focus on modularity and scalability, utilizing exclusively open-source technologies such as the Gazebo simulator Robot Operating System (ROS) as its foundational components because in this competition simulator is preferred over physical testing due to its cost-effectiveness, safety, accessibility, rapid iteration, scalability, controlled environment, data collection, and time efficiency advantages [5]. There is a diverse range of algorithms employed in the domain of autonomous racing. These cutting-edge algorithms encompass perception, planning, and control strategies that enable autonomous vehicles to navigate racetracks with precision

and speed. These algorithms must be capable of making high level decisions, selecting optimal racing lines, and executing manoeuvres while ensuring safety and efficiency.

Employing point-to-point trajectory planning, the author guided the ego vehicle for optimal racing while prioritizing safety, resulting in a 3rd place collision-free finish in the IAC, just 0.44 seconds behind the winner [6]. A new LiDAR-based algorithm distinguishes curved and straight track walls, seamlessly integrating preprocessing with driving tasks. It employs clustering for object detection and achieved a 0.11-meter RMS lateral offset error across 200 data frames [7]. As compared to traditional SLAM methods, Dynamic Deep Learning SLAM (DDL-SLAM) addresses occlusion, robustness, and stability issues in dynamic environments. Experiments showed DDLSLAM significantly outperformed ORB-SLAM2, reducing S.D. and RMSE by 98.2% and 98.6% , respectively [8].

Planning determines the trajectory and manoeuvres of the racing vehicle. The planning algorithm is responsible for making high-level decisions, such as selecting the best racing line, determining when to accelerate, brake, or steer, and calculating the optimal path to follow on the racetrack. An innovative approach to trajectory planning for autonomous racing vehicles utilizes mixed-integer optimization (MIO). This method efficiently handles obstacles and rewards, achieving computation times as low as 0.4 seconds in Bedford race circuit simulations. [9]. A game-theoretic planner (GTP), enables the vehicle to strategically predict other vehicles' behavior to avoid collisions and advance on the track. The GTP car wins 98% of the time by planning its trajectory ahead of the MPC car, blocking its path. However, it requires the trajectory to be represented as a piecewise polynomial, which may restrict its applicability in certain scenarios [10]. Another approach employs embedded nonlinear model predictive control (NMPC) input-output data to train an ANN controller for miniature autonomous vehicles (AVs).Over the course of 5000 epochs, the training culminated in an impressive 98.7% overall training accuracy [11]. Enhancing driving control for vision based-based autonomous driving, authors introduce deep-MCTS. Two neural networks predict actions and probabilities for multiple trajectories, thereby improving steering control stability training efficiency, driving trajectory stability by 66.30%, 50% and 59.06% respectively [12].

The integration of deep learning has been a game-changer, improving the efficiency, safety, and adaptability of autonomous vehicles on the racetrack. Reinforcement learning agents learn from trial and error, iteratively refining their control strategies to achieve high-speed racing while adhering to safety constraints. Viability Theory-based supervision ensures safe training, preventing crashes and maintaining friction limits. While minimal risk planners have increased likelihood of success, ordinary planners are quicker on every trial map. In a particular test case, 100% success rate is achieved by the minimal risk agent while a traditional planner achieves 45% success rate. [13]. Another deep reinforcement learning system achieves superhuman performance in Gran Turismo Sport by defining a reward function and using an off-policy training having an experience replay buffer. Using two different cars on the same track, it outperforms the quickest lap time set by humans by 0.15 and 0.04 seconds, respectively. [14]. The Trajectory-aided Deep Reinforcement Learning (TAL) approach improves DRL agent performance in high-speed autonomous racing. At 4 m/s, the baseline agent's completion rate is 100%; at 6 m/s, it declines to 50%.On the other hand, the completion rates of TAL agents are higher; the TAL planner having a speed of 6 metres per second reaches 60%, while the agent having a speed of 8 metres per second reaches 40%. [15]. By leveraging the vast capabilities of deep learning, the field of autonomous racing is advancing towards more sophisticated and competitive autonomous racing systems.

2. Objectives

The primary objective was to employ a novel lidar downscaling approach for efficient RL training. The aim was to preserve the 'Key Points' of the LIDAR data while reducing the overall sample size to a fixed length. This algorithm was planned for integration into a ROS2 Node to modify the LIDAR scan. The integration was targeted to work within the F1-Tenth Gym ROS simulator and Gymnasium Environment for Reinforcement Learning.

The second goal was to develop an autonomous navigation algorithm by utilizing reinforcement learning techniques involving creation of a customized Gymnasium Environment for integration with Stable Baselines 3 and ROS2 (Robot Operating System 2) on Ubuntu 20.4 with necessary dependencies installed.

The third objective aimed at developing a planning and decision-making module for optimal racing trajectory generation using Stable Baselines 3 for model training with parameter tuning and reward policy.The goal here was to optimize reward policy to motivate survival time and collision avoidance.

3. Methods

The designed algorithm aims to downscale LiDAR data empowering the race car to make informed decisions in the F1/10th Simulator. The racecar's abilities are enhanced using Reinforcement Learning (RL). The racecar is considered the agent, and the racetrack and surroundings are the environment. Based on the agent's interaction with the environment, it learns to make optimal decisions to maximize a predefined reward.

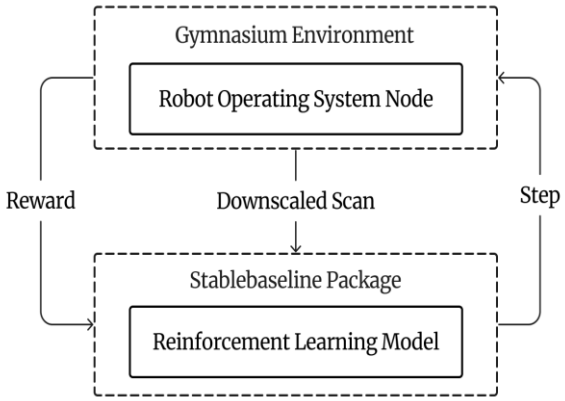

Fig. 1: Framework of simulation environment

Subscribing to topics is pivotal for enabling real-time communication and data processing within the F1/10th Simulator in ROS2 (Robot Operating System). This was achieved by initializing nodes with rclpy, setting up data processing callbacks, and creating subscriptions through 'create subscription' which ensured nodes maintained responsiveness and proficiently managed data exchanges within the simulator's dynamic environment.

A. LiDAR Data Down-scaling

F1 Tenth Gym simulator enables a LiDAR scan comprising 1080 data points, aiding in understanding the vehicle's environment. However, to optimize RL's data-intensive nature, downscaling LiDAR data is essential. This downscaling can be achieved through sampling at a fixed rate given by eq. (1) to extract 20 of 1080 points, though it may lack adaptability. A more dynamic approach, feature selection relies on pre-labeled data but introduces potential bias.

$$\mathrm{DS} = \{data[i \cdot 54] | i \in \mathbf{N}\} \qquad (1)$$

A Piecewise-Linear Function Approximation method using a fixed number of lines is used to tackle this. This technique guarantees a minimal root mean square error [16]. In practical terms, processing a dataset of 1080 points with the Python implementation of the algorithm took around 9.87 seconds, resulting in 10 optimal line segments. To achieve a manageable processing time of approximately 0.093 seconds, the algorithm sampled every 10th data point before application.

A novel data downscaling algorithm proposed in this paper having linear time complexity, also incorporates the addition of points with the minimum distance value in both the first and second halves of the LiDAR data. This provides the necessary information for reinforcement learning (RL) application to avoid collisions within its environment.

A reference value threshold needs to be tuned to take into account noise in the data. Let C be a set of all critical points given by eq. (2)

$$C = \left\{ \begin{array}{l} (j, data[j]) \,|\, \forall i, j, k \text{ where } i \neq j \neq k : \\ |data[i] - data[j]| > \text{threshold}, \\ |data[j] - data[k]| > \text{threshold}, \\ \text{sgn}(data[i] - data[j]) \neq \\ \text{sgn}(data[j] - data[k]) \end{array} \right\} \quad (2)$$

The following algorithm is applied to get the final downscaled data where n is the number of original data points.

Algorithm 1 Downscaling to k points

Input: Data sequence, k, threshold
Output: Critical points
Initialization $diff\_list \leftarrow [], i \leftarrow 0, n, threshold$
while $i < n - 2$ **do**
  $j \leftarrow i + 1$
  $diff \leftarrow data[i] - data[j]$
  if $|diff| > threshold$ **then**
    $k \leftarrow j + 1$, $diff\_2 \leftarrow data[j] - data[k]$
    while $k < n - 1$ **and** $|diff\_2| > threshold$ **and** $diff\_2 \cdot diff > 0$ **do**
      $k \leftarrow k+1, j \leftarrow j+1, diff\_2 \leftarrow data[j] - data[k]$
    end while
    Append $[j, |data[j] - data[i]|]$ **and** $[i, |data[j] - data[i]|]$ to $diff\_list$
    $i \leftarrow j$
  end if
  $i \leftarrow i + 1$
end while
Sort $diff\_list$ based on the second element in descending order
Append min value points in the first and second halves of data to $critical\_points$
Add the first $k-2$ points of $diff\_list$ to $critical\_points$
return $critical\_points$

B. System Design and Implementation

The project utilized the F1 Tenth Gym simulator with consistent topic names. ROS2 nodes are structured for rapid planning algorithm changes. This control system issued Ackerman drive commands to control speed and steering angle.

In this study, a custom Gym environment was crafted, using OpenAI's Gymnasium interface. This environment acts as a vital bridge between the Reinforcement Learning agent and the F1 Tenth Gym Simulator enabling the issuance of control commands for steering and speed, along with resetting the car's initial position. It offers a structured platform for the RL agent to execute actions and receive LiDAR data observations.

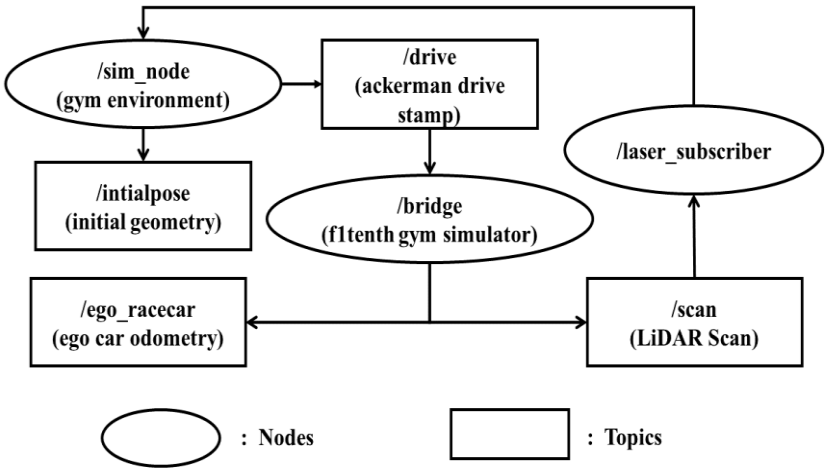

Fig. 2: Robot Operating System Communication Control

For the training of RL models, the project harnesses the Stable Baselines library, particularly employing the Proximal Policy Optimization (PPO) algorithm and Multi Layer Perceptron (MLP) policy. During this training phase, the RL model learns to establish associations between LiDAR observations and optimal control actions, progressively enhancing its competence in autonomous navigation.

The environment defines the action space and observation space for the reinforcement learning agent: Action Space for the reinforcement learning agent is discrete and consists of three possible actions that the agent can take, namely maintain the current state, accelerate or change steering action, corresponding to different control commands for the car. Similarly, the observation space is represented as a vector with 10 values, which corresponds to distances obtained from downscaled LiDAR sensor data and the reward policy correlates directly with the time the vehicle operates without experiencing collisions, motivating the agent to prioritize collision avoidance and efficient navigation. Reward is calculated using the following equation:

$$\text{reward} = \text{prev reward} + (10 - \text{action}) - 3(\text{min distance}) \quad (3)$$

This equation takes the reward achieved in the previous episode and a baseline reward of 10 into account and penalizes the agent for its chosen action, where 'action' denotes the selected action. The penalty is if the agent takes a turn (left or right), promoting the agent to go straight for optimal rewards. Similarly, the agent is penalized based on its proximity to a wall, with 'min distance' representing the distance from the wall. The penalty scales with three times the distance, motivating the agent to keep a safe distance from the wall thus discouraging collision. Following training, the RL model takes charge of autonomous vehicle control, using LiDAR data and acquired policies to steer the car in the simulator.

4. Results

The study conducted on the ROS-based F1 TENTH Gym employed three distinct LiDAR scan data downscaling approaches: normal sampling, piecewise linear approximation, and a novel downscaling method. These methods were thoroughly analyzed and were integrated as preprocessing steps to downscale LiDAR data, serving as inputs for the subsequent training of our reinforcement learning model.

Table 1 presents the computation time taken for downsizing data, while Table 2 displays the root mean square error of downscaled data. The mean square error of downscaled data is presented in Table 3.

TABLE I: Computation Time Taken to Downscaled Data

Approach	Setting 1	Setting 2	Setting 3
Novel downscaling approach	0.000506 s	0.000403 s	0.000409 s
Polyline approximation	0.092464 s	0.090552 s	0.094992 s
Normal sampling	0.000013 s	0.000014 s	0.000015 s

TABLE II: Root Mean Square Error of Downscaled Data

Approach	Setting 1	Setting 2	Setting 3
Novel downscaling approach	0.70	0.51	0.49
Polyline approximation	0.41	0.37	0.28
Normal sampling	0.36	0.44	0.22

TABLE III: Mean Square Error of Downscaled Data

Approach	Setting 1	Setting 2	Setting 3
Novel downscaling approach	0.49	0.26	0.24
Polyline approximation	0.17	0.14	0.08
Normal sampling	0.13	0.19	0.05

5. Discussion

Normal sampling resulted in the preservation of only one peak while failing to retain any valley points in Fig. 3 Setting A. In contrast, both the polyline approximation and the novel approach successfully preserved two out of three peaks and a single valley point. The novel downscaling approach excelled in maintaining the overall structural integrity of the graph. Comparable patterns were noted for Fig. 3 Setting B and Fig. 3 Setting C. In both cases, the novel downscaling approach excelled in preserving the overall shape of the graph.

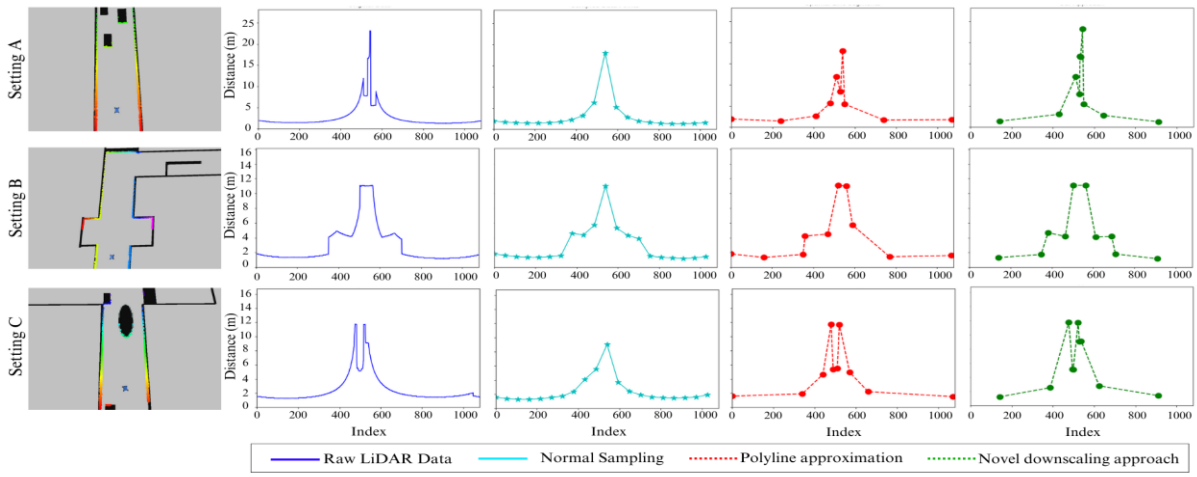

Fig. 3: Comparative Analysis of Data Downscaling Techniques

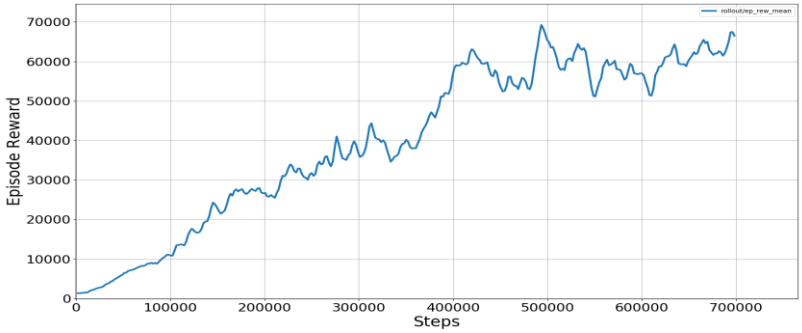

Fig. 4: Mean episode length over steps

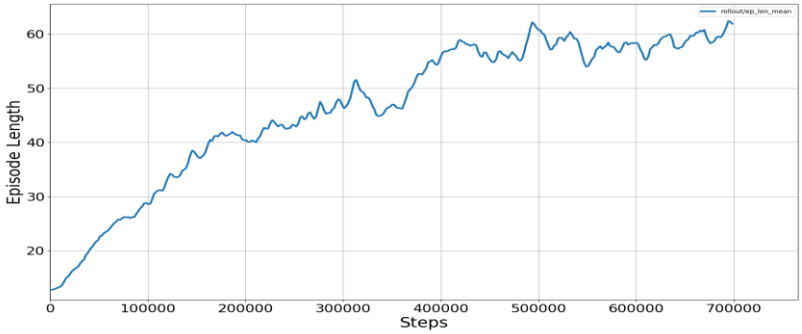

Fig. 5: Mean episode reward over steps

The objective of the down-scaling techniques was to determine their efficacy in retaining vital data features. The novel downscaling approach proposed in this paper is approximately 217 times faster than polyline approximation when used on LiDAR data with 1080 points. Despite having a larger root mean squared and mean squared error , the novel method is superior in terms of computational efficiency and its ability to maintain the overall graph shape while preserving essential features. However during training, the RL model did not reach convergence even after 700 thousand timesteps. The best trained model achieved the episode reward of 91,856 with episode length of 73 on the 493.6 thousandth time step shown in Fig. 4 and 5 .

The trained model when tested on the same race track completed 80% of the track length. On testing the model on different tracks it was observed, the agent traversed the straight path effectively but struggled with sharp curves. The proposed research developed an autonomous navigation algorithm for efficient use in F1/10 racing scenarios. The novel LiDAR downscaling technique was tested and proven to maintain the shape and essential features of the data. Another unique feature was the implementation of a custom Gym environment designed for reinforcement learning. Rapid integration of these components was made possible through the use of ROS2, which enabled development and simulated testing. Despite the advantages of this approach, such as improved efficiency, accelerated development, and robust testing in a simulated environment, the RL model did not converge even after 700 thousand time steps. Nevertheless, it was still able to effectively navigate through straight patches and gentle curves. To enhance the system's reliability and prepare it for real-world application on actual vehicles, additional improvements are essential. Future scope will focus on fine-tuning the reinforcement model to further enhance the algorithm's performance.

References

[1] B. Padmaja, CH. V. K. N. S. N. Moorthy, N. Venkateswarulu, and M. M. Bala, "Exploration of issues, challenges and latest developments in autonomous cars," Journal of Big Data, vol. 10, no. 1, May 2023,.

[2] B. Li et al., "From Formula One to Autonomous One: History, Achievements, and Future Perspectives," in IEEE Transactions on Intelligent Vehicles, vol. 8, no. 5, pp. 3217-3223, May 2023.

[3] J. Betz et al., "Autonomous Vehicles on the Edge: A Survey on Autonomous Vehicle Racing," in IEEE Open Journal of Intelligent Transportation Systems, vol. 3, pp. 458-488, 2022.

[4] B. Li et al., "From Formula One to Autonomous One: History, Achievements, and Future Perspectives," in IEEE Transactions on Intelligent Vehicles, vol. 8, no. 5, pp. 3217-3223, May 2023.

[5] V. S. Babu and M. Behl, "f1tenth.dev - An Open-source ROS based F1/10 Autonomous Racing Simulator," 2020 IEEE 16th International Conference on Automation Science and Engineering (CASE), Hong Kong, China, 2020, pp. 1614-1620.

[6] G. Hartmann, Z. Shiller and A. Azaria, "Competitive Driving of Autonomous Vehicles," in IEEE Access, vol. 10, pp. 111772-111783, 2022.

[7] S. W. Meyer and D. M. Bevly, "Cluster-Based Wall Curvature Detection and Parameterization for Autonomous Racing using LiDAR Point Clouds," IFAC-PapersOnLine, vol. 55, no. 37, pp. 494–499, 2022.

[8] Y. Ai, T. Rui, M. Lu, L. Fu, S. Liu and S. Wang, "DDL-SLAM: A Robust RGB-D SLAM in Dynamic Environments Combined With Deep Learning," in IEEE Access, vol. 8, pp. 162335-162342, 2020.

[9] R. Reiter, M. Kirchengast, D. Watzenig, and M. Diehl, "Mixed-integer optimization-based planning for autonomous racing with obstacles and rewards," IFAC-PapersOnLine, vol. 54, no. 6, pp. 99–106, 2021.

[10] M. Wang, Z. Wang, J. Talbot, J. C. Gerdes and M. Schwager, "Game-Theoretic Planning for Self-Driving Cars in Multivehicle Competitive Scenarios," in IEEE Transactions on Robotics, vol. 37, no. 4, pp. 1313-1325, Aug. 2021.

[11] A. Tătulea-Codrean, T. Mariani, and S. Engell, "Design and Simulation of a Machine-learning and Model Predictive Control Approach to Autonomous Race Driving for the F1/10 Platform," IFAC-PapersOnLine, vol. 53, no. 2, pp. 6031–6036,2020.

[12] J. Chen, C. Zhang, J. Luo, J. Xie and Y. Wan, "Driving Maneuvers Prediction Based Autonomous Driving Control by Deep Monte Carlo Tree Search," in IEEE Transactions on Vehicular Technology, vol. 69, no. 7, pp. 7146-7158, July 2020.

[13] B. D. Evans, H. W. Jordaan, and H. A. Engelbrecht, "Safe reinforcement learning for high-speed autonomous racing," Cognitive Robotics, vol. 3, pp. 107–126, Jan. 2023.

[14] F. Fuchs, Y. Song, E. Kaufmann, D. Scaramuzza, and P. Duerr, "Super-Human Performance in Gran Turismo Sport Using Deep Reinforcement Learning," IEEE Robotics and Automation Letters, pp. 1–1, 2021.

[15] B. D. Evans, H. A. Engelbrecht and H. W. Jordaan, "High-Speed Autonomous Racing Using Trajectory-Aided Deep Reinforcement Learning," in IEEE Robotics and Automation Letters, vol. 8, no. 9, pp. 5353-5359, Sept. 2023.

[16] E. Camponogara and L. F. Nazari, "Models and Algorithms for Optimal Piecewise-Linear Function Approximation," Mathematical Problems in Engineering, vol. 2015, p. e876862, Jul. 2015.

[17] O'Kelly, Matthew and Zheng, Hongrui and Karthik, Dhruv and Mangharam, Rahul, "F1TENTH: An Open-source Evaluation Environment for Continuous Control and Reinforcement Learning," NeurIPS 2019 Competition and Demonstration Track, pp. 77-89, 2020.

[18] Raparthi, M., Dodda, S. B., & Maruthi, S. (2023). Predictive Maintenance in IoT Devices using Time Series Analysis and Deep Learning. Dandao Xuebao/Journal of Ballistics, 35(3). https://doi.org/10.52783/dxjb.v35.113