engineering proceedings

MDPI

Proceeding Paper

Reinforcement Learning to Calculate Routes for Simulated Robot Safety Cones†

Eva Murio [1*], Jesús Balado [1] and Pedro Arias [1]

[1] GeoTECH Group, CINTECX, Universidad de Vigo, 36310 Vigo, Spain
* Correspondence: eva.muro@yahoo.com
† Presented at the 4th International Electronic Conference on Applied Sciences, 27 October–10 November 2023; Available online: https://asec2023.sciforum.net/.

Abstract: This paper presents a model of a cone using reinforcement learning, harnessing the self-learning capacity of Artificial Intelligence to improve process efficiency. The independent operation of the cone is achieved through a reward and punishment system based on approaching or reaching the goal. The cone must decide between 0º or 90º turns at each step to maximize long-term rewards. While the simulated robotic safety cones successfully reach their targets, the training process is time-consuming due to the numerous variables involved. Nonetheless, the rise of AI and its self-learning capabilities offer promising opportunities for process optimization.

Keywords: Reinforcement Learning; pathfinding; simulation; road environment; work zone.

Citation: Murio, E.; Balado, J.; Arias, P. Reinforcement Learning to Calculate Routes for Simulated Robot Safety Cones. *Eng. Proc.* **2023**, *52*, x. https://doi.org/10.3390/xxxxx

Academic Editor(s):

Received: date
Revised: date
Accepted: date
Published: date

1. Introduction

The significance of transportation, particularly in the context of road maintenance and construction, cannot be overstated [1]. Despite the evident advantages of automation, this industry has been slow to adopt modernized tools and procedures [2]. Embracing automation in road construction can yield numerous benefits, including enhanced efficiency, reduced physical strain on workers, shortened construction timelines, and minimized economic losses. In the road construction setting, traffic cones play a pivotal role in delineating work zones. Traditionally, these cones are manually placed and relocated as the project progresses [3]. However, by introducing automation, this process can be significantly expedited, thereby enabling workers to focus on more intricate tasks. Conventional robotic systems often require an operator to control their actions, thereby limiting the potential efficiency gains [4]. To overcome this limitation, we propose a solution centered around an autonomous robot capable of independently reaching the desired position through the utilization of reinforcement learning techniques.

The power of Reinforcement Learning (RL) has been exemplified through various instances in the field of control and robotics, as demonstrated in [5]. These examples encompass a broad range of applications, including a two-armed robot mastering the art of juggling, a mobile robot efficiently pushing boxes over extended periods, and the coordinated collection and transportation of disks by multiple robots to predetermined destinations. Despite the numerous advantages offered by RL, the authors acknowledge the challenges associated with its design, primarily due to its reliance on a trial-and-error approach. To facilitate the development of future works, the authors propose several ideas. These include decomposing complex problems into smaller, more manageable subproblems, and providing continuous reward signals to the RL agent to enhance feedback and learning efficiency.

The paper [6] provides a comprehensive analysis of the application of Deep Reinforcement Learning (DRL) in mobile robot navigation between 2016 and 2021. The review highlights the limitations of traditional methods in handling unknown or dynamic

environments and emphasizes how RL can address these challenges. Notably, RL's advantages, including its reduced reliance on sensor accuracy, self-learning capabilities, and real-time recalculations, are discussed. The authors' paper also presents specific points that are relevant to the design of the state space model in mobile robot navigation. These include the representation of the starting and target points using the current and destination coordinates of the mobile robot. Furthermore, the use of simplified 2D models is favored over complex 3D models due to their simplicity, which aligns with the implementation choices made in this project.

For the development of current work, the following specific objectives have been defined:

- Construction of a dynamic-kinematic model for a robotic safety cone.
- Establishment of a suitable agent according to type of spaces for action and observation and the requirement of neural networks.
- Development of a reward function that allows achieving good results during the shortest training times.

2. Method

The basic elements of reinforcement learning include an agent block, an environment block and action, observation, and reward signals. The agent, who is surrounded by the environment, interacts with it by performing actions and receives feedback through the observation and reward signals. This scheme is constituted by the RL basic elements and the necessary subsystems (Figure 1). The traffic cone is based on a differential robot model. It is the environment and is responsible for sending the 2D position and the rotation angle θ concerning the positive x axis. The agent chooses the perfect combination of angles θ to reach the desired position using the action signal.

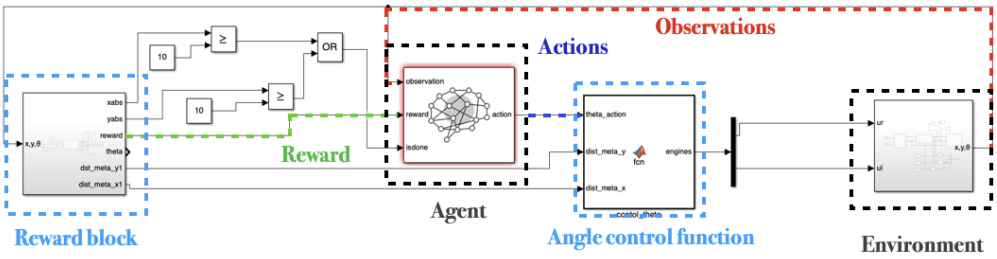

Figure 1. General scheme.

2.1. Agent

Due to the complexity of the process, an AC agent is elected. It is composed of two neural networks: the critic and the actor (Figure 2). The actor uses a politic gradient to estimate the probabilities of taking each possible action from each state and, therefore, chooses the action that maximizes the long-term reward. This action is evaluated by the critic network using a value or action-value function, which estimates the long-term reward you will receive from each state or state-action. The value function from the critic is compared with the current value function from the environment and, as a result, the error is calculated. This error will feed back to the actor and critic to improve decision making.

The summary of the features of the agent is:

- Observations: 2D robot position and turning angle θ relative to the positive x-axis.
- Observation space: continuous.
- Action: turning angle θ.
- Action space: discrete, with angles of 0º or 90º.
- Agent: AC type. It contains an actor-critic algorithm. Suitable for continuous or discrete action spaces.

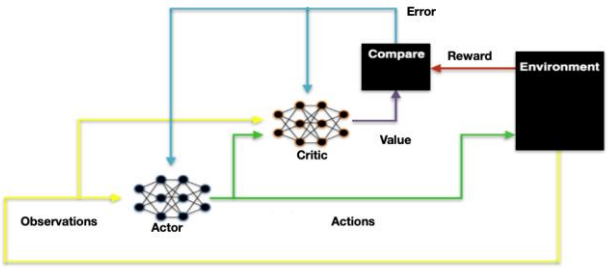

Figure 2. Actor-critic scheme.

2.2. Reward block

The reward block consists of two constants defining the 2D target position, a Calculation function, and a reward design subsystem (Figure 3). The Calculation function obtains the difference between the target position and the current position, sets an 3% error for the goal, and applies it to the signals mepase and fin. The mepase signal indicates if the goal has been exceeded or if the cone is exploring a wrong quadrant and the fin signal indicates if the goal has been reached.

The reward design is based on the following criteria:

- Reward
- +1 when the cone approaches the goal.
- + 100 when the cone reaches the goal.
- Penalty
- -5 when the cone moves away from the goal.
- -30 when the cone exceeds the goal or is exploring a wrong quadrant.
- IsDone condition
- When the cone exceeds ± 10.

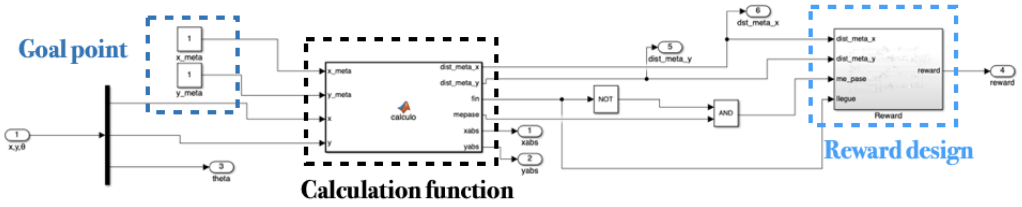

Figure 3. Reward block scheme.

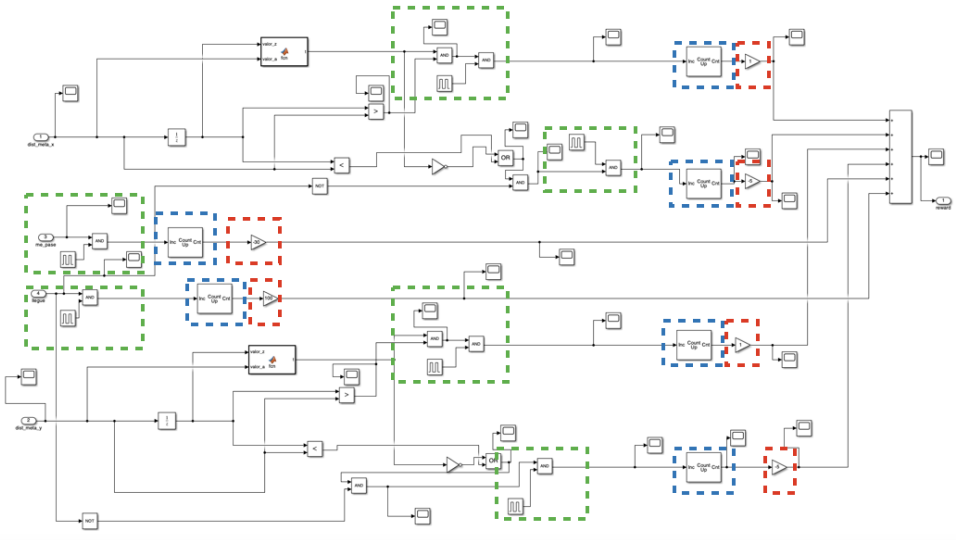

Figure 4. Reward design subsystem.

Within the reward design subsystem (Figure 4), a pattern of parallelism emerges in each signal. First, the digital signal is compared with a pulse signal to utilize the constant reward concept. Then, this signal is passed through a rising edge counter and multiplied by its corresponding value defined in the criteria. In the case of distances to the goal, it is proceeded to compare a current value with the previous one by applying an *UnitDelay*. If the previous value is greater than the current one, the cone is approaching the goal and it is awarded. Otherwise, it is penalized.

2.3. Environment

The environment is based on the dynamic-kinematic model of a differential robot, where the inputs are the engines, and the outputs are the 2D position and the turning angle. The robot is composed of a rigid body with two independent wheels and a castor wheel that provides stability. To develop the scheme, the following steps were taken: solving kinematics, solving dynamics, obtaining the model in state variables, and simulating it.

The scheme (Figure 5) consists of a block that implements the state variables model for the dynamic-kinematic relationship. This block outputs linear velocity and turning angle, allowing us to calculate the velocity of each axis. By integrating the velocity, the position is determined. Additionally, there is a draw function in charge of simulating the movement of the robot. The scheme also establishes initial constants so that the robot returns to its starting position after each training episode. In our case, the starting position is (0, 0, 0).

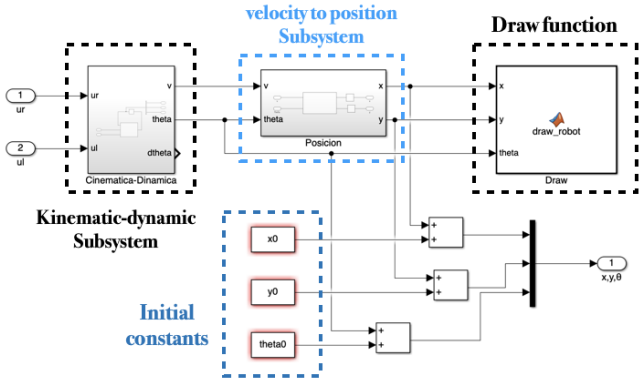

Figure 5. Dynamic-kinematic model of a differential robot.

2.4. Angle control function

This function converts the angle provided by the agent, through the dynamics and kinematics equations of the differential robot, into the corresponding motor signals. It has been decided to do it this way, instead of the action being the motors directly, because the agent only has one variable to discover, which will make things easier. This function defines a speed, that varies around 1 m/s, and choose at what moment the motors must stop, due to the existence of a non-rectilinear deceleration that must be considered, otherwise the goal will not be reached. So, to configure the distances in which they will have to be stopped, the model must first be simulated with Signal Builder.

3. Results

After of the creation of the environment and its corresponding observation and action specifications, the reset function is established. It is activated when the stipulated time of

each training episode runs out or if the *IsDone* condition is reached. Afterwards, the AC agent and its neural networks are defined and finally, the model is trained.

The proposed method was implemented in MATLAB_R2022b. The training was conducted in Reinforcement Learning Episode Manager, which shows a graph with the reward of each episode and its average reward throughout the training. This tool displays a graph where the x-axis represents the number of episodes, the y-axis represents the episode reward and the light and dark blue lines represent the reward for each episode and the average reward, respectively. As shown in the Figure 6, the first time the robot reached the goal was in episode 3, taking 39 min 26 s. Despite this, the robot continued exploring the area periodically using the exploration vs. exploitation concept. Then, the cone went more frequently towards the target, as seen in episodes 32, 33 and 34 or 40, 41 and 42, until the cone went constantly, as shown around episode 60. Due to the continuous recurrence of the robot going to the point, we concluded the training in the episode 113 in a time of 22h 22min 49s.

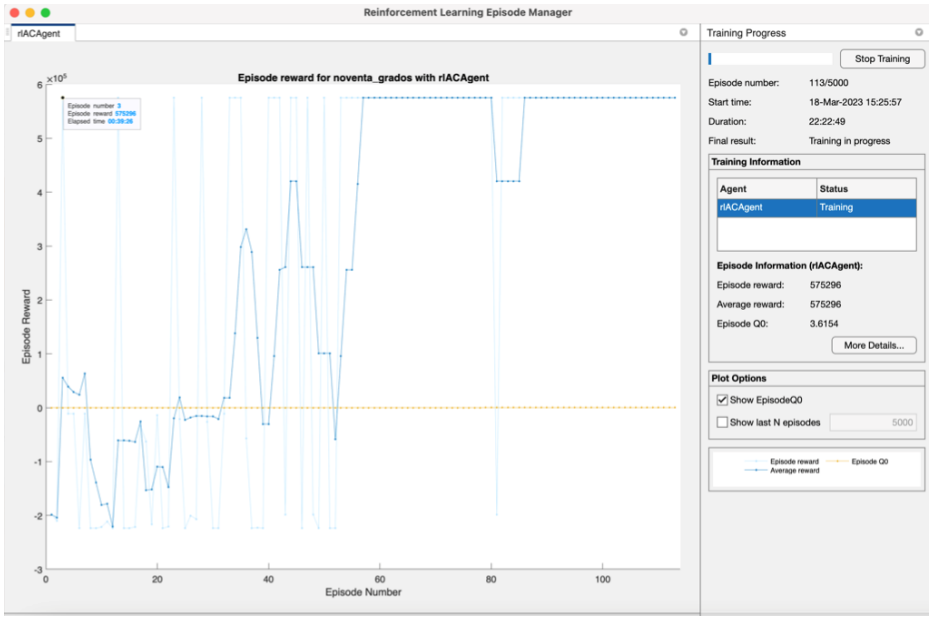

Figure 6. Reinforcement Learning Episode, point (1,1).

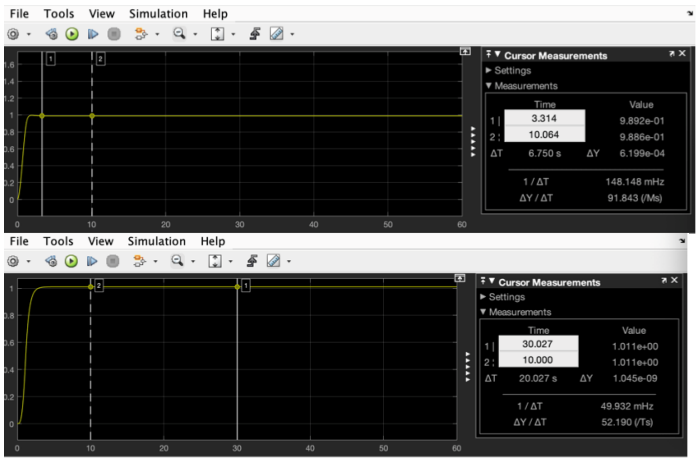

Figure 7. Graphs corresponding to the final position x (up) and y (down) of the robot's mass center.

As shown in Figure 7, the point lied between the established limits 1.03-0.97 m. This was caused by the previous examination of the kinematic-dynamic model of the robot using Signal Builders, in which it was displayed how to reach the goal considering the

moment where the power supply to the engines is stopped. Consequently, it was possible to calculate the distance that the cone moves in the deceleration. This distance changes depending on the movement that the robot is making, so it is not unique, and it will have to be changed and recalculated by the previous testing with Signal Builders when the target-point varies. The position of the robot and the start-target points is shown in Figure 8.

4. Conclusion

This work demonstrates the feasibility of attaining a predetermined objective through the successful implementation of a robotic system with Reinforcement Learning (RL). This achievement indicates a notable enhancement in operational efficiency within real-world logistical processes. Moreover, RL is easily implementable with regards to security measures to ensure the physical integrity of the robot's body remains intact.

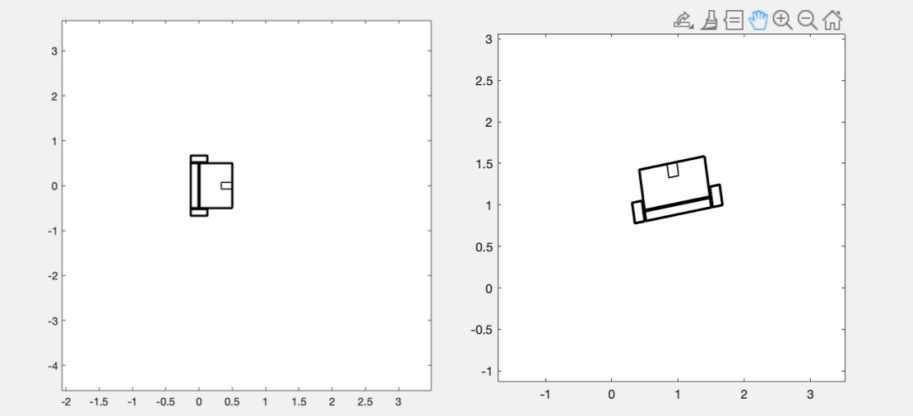

Figure 8. Start point (left) and target point (right) of the robot using draw_robot function.

Throughout this work, numerous challenges had surfaced. Maintaining a constant speed has proven to be unfeasible, as it introduces an additional variable that complicated the training process. Another obstacle encountered was the deceleration of the robotic cone, necessitating prior modeling of motor movements. Future work will focus on: (1) identifying an equation that describes the deceleration of the cone, (2) exploring the feasibility of achieving a constant speed, and (3) extending the implementation to encompass the mapping signal for all quadrants, since currently, the work focused on a single point within the positive (x, y) plane.

Author Contributions: Conceptualization, J.B.; methodology, E.M.; software, E.M.; validation, E.M., Y.Y. and Z.Z.; resources, P.A.; writing—original draft preparation, E.M.; writing—review and editing, J.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research has received funding from Xunta de Galicia through human resources grant (ED481B-2019-061) and GAIN (grant number ED431F 2022/08). This paper was carried out in the framework of the InfraROB project (Maintaining integrity, performance and safety of the road infrastructure through autonomous robotized solutions and modularization), which has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 955337. It reflects only the authors' views. Neither the European Climate, Infrastructure, and Environment Executive Agency (CINEA) nor the European Commission is in any way responsible for any use that may be made of the information it contains.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Yarmukhamedov, S., Smith, A. S., & Thiebaud, J. C. (2020). Competitive tendering, ownership and cost efficiency in road maintenance services in Sweden: A panel data analysis. Transportation Research Part A: Policy and Practice, 136, 194-204.

2. Li, J., Yin, G., Wang, X., & Yan, W. (2022). Automated decision making in highway pavement preventive maintenance based on deep learning. Automation in Construction, 135, 104111.
3. Katsamenis, I., Bimpas, M., Protopapadakis, E., Zafeiropoulos, C., Kalogeras, D., Doulamis, A., ... & Lopez, R. (2022, June). Robotic maintenance of road infrastructures: The heron project. In Proceedings of the 15th International Conference on PErvasive Technologies Related to Assistive Environments (pp. 628-635).
4. Wang, F., Dong, W., Gao, Y., Yan, X., & You, Z. (2019). The Full-Automatic Traffic Cone Placement and Retrieval System Based on Smart Manipulator. In CICTP 2019 (pp. 3442-3453).
5. Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey. Journal of artificial intelligence research, 4, 237-285.
6. Zhu, K., & Zhang, T. (2021). Deep reinforcement learning based mobile robot navigation: A review. Tsinghua Science and Technology, 26(5), 674-691.