2023 Latin American Robotics Symposium (LARS), 2023 Brazilian Symposium on Robotics (SBR), and 2023 Workshop on Robotics in Education (WRE) | 979-8-3503-1538-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/LARS/SBR/WRE59448.2023.10332988

SLAYO-RL: A Target-Driven Deep Reinforcement Learning Approach with SLAM and YoLo for an Enhanced Autonomous Agent

José Montes
Universidade Federal Fluminense
Niterói, Rio de Janeiro, Brazil
joselucasbrandaomontes@id.uff.br

Troy Costa Kohwalter
Universidade Federal Fluminense
Niterói, Rio de Janeiro, Brazil
troy@ic.uff.br

Esteban Clua
Universidade Federal Fluminense
Niterói, Rio de Janeiro, Brazil
esteban@ic.uff.br

***Abstract*—This article presents an innovative approach for training an agent to reach a specific and predetermined target in an unknown environment. It uses reinforcement learning for an agent with a Lidar sensor and a camera. Given the difficulty of using raw high-dimensional information to train any reinforcement learning agent, the Lidar sensor data was processed using Simultaneous Localization and Mapping to provide the agent's location in space. To identify the agent's target of interest, the camera image was processed using the YoLo object detection model to provide the coordinates of the target in the image. In addition to processing the agent's state, the two technologies were used as a composition of the reward obtained by the agent, causing it to develop the behavior of exploring an unknown environment and, after locating the target, moving towards it until the agent collides with the target. The proposed approach differs from the state of the art because it unites the two technologies as a composition of the agent's state and reward.**

***Index Terms*—SLAM, YoLo, Deep Reinforcement Learning**

I. Introduction

Using techniques based on Artificial Intelligence is increasingly present in developing solutions for various robotics problems, making it possible to highlight the field of autonomous robotics, which has several problems related to movement, navigation, and interaction with the environment.

Numerous advancements in technology have led to significant progress in enabling robots to navigate and interact seamlessly within intricate environments. One particularly crucial development is the availability of Digital Twins, which has spurred the growth of various other technologies through the virtual simulation of real agents. Since the cost of training this type of solution in a real environment is very high, researchers use virtual training environments instead [1], highlighting the Reinforcement Learning approach [2], mainly using a target-driven approach [3].

Reinforcement learning is presented as an AI technique where an agent interacts with an environment and collects information, using trial-and-error experience [4] that benefits from the use of computer simulation to build solutions in robotics. However, this strategy requires a very large amount of interactions with the environment, which becomes unfeasible in a real environment.

The use of reinforcement learning, despite representing an important advance in problem-solving in the field of robotics [4], still presents some difficulties in the interpretation of raw data, such as data coming from sensors and information about the environment where the robot is inserted. It is possible to use different techniques for processing raw and pre-processed information together with reinforcement learning. Among the existing technologies that can interpret the raw data coming from sensors in a robot, solutions for mapping and localization, and identification of objects in images can be highlighted.

Target-driven Reinforcement Learning is remarked in the work of Ruan *et. al.* [5] as a type of machine learning that is based on the training of an agent to reach a certain target or specific objective, where this behavior can be achieved based on specified targets and corresponding rewards in the environment. For the mapping and localization problem, it is possible to use SLAM (Simultaneous Localization and Mapping) [6] [7] to allow the robot to build a map of the surrounding environment while maintaining its bearing within this map. Once the problem of localization and mapping is solved, it is still necessary for the robot to be able to identify and locate the target to be reached, a task that can be solved using the YoLo deep learning model [8] [9], a solution for detecting and identifying objects in images, which enables the robot to identify its target in the environment.

The combination of these technologies has the potential to result in the construction of an improved control solution for autonomous robotics since SLAM provides location coordinates on the created map and YoLo provides target identification in the environment. Both provide high-level information, which, combined and used in training the reinforcement learning agent, can make it learn to move through a simulated environment, enabling training in various interactions with the environment. Despite the possibilities obtained through this integration, the works related to reinforcement learning enable the integration of only one of these technologies with the agent training process.

Motivated by these questions, our proposed work integrates target-directed reinforcement learning with information from SLAM and YoLo for the task of locating a target in an unfamiliar environment It explores the potential benefits of this approach for autonomous robotics. As a result, we present an approach for integrating these technologies in training a reinforcement learning agent to control a simulated robot. We demonstrate the results by separately comparing scenarios using SLAM and YoLo with our integrated approach in agent training. Our main contribution is to present an integrated solution to the problem of locating a target in an unknown environment using reinforcement learning and taking advantage of SLAM mapping and YoLo object detection.

Our implementation was based on the construction of a simulated environment using the Unity3D [10] development environment, a game development tool that, through the ML-Agents library [11] enables integration with a reinforcement learning agent from the library itself or an external one, through communication through Python tools, alternative chosen in the development of the present work. For the construction and training of the agent, the Stable-Baselines 3 library [12] was chosen because, in initial tests with other libraries, it was the one that presented the best results. For the implementation of SLAM, the BreezySlam library [13] was used due to its ability to integrate with real sensors. For the object detection task, the YoLov5 library [14] was adopted.

This article is organized as follows: Section 2 presents works related to the research problem addressed in this work and the gaps that reside when using SLAM mapping and object detection to solve a single problem. Section 3 presents our proposal. Section 4 presents details of the implementation of our solution and the experiment results. Section 5 presents the conclusion of the present work.

II. Related Work

SLAM mapping is used in several published works related to robotics. However, this method has only recently been successful in integrating with reinforcement learning, largely due to computational advances and deep learning techniques [4]. Among the existing works, it is possible to highlight Zheng *et. al.* [15] that implements a hierarchical exploration strategy for completing autonomous exploration tasks in environments with crowds. The exploration strategy includes a global planner based on a Travel-Salesman-Problem (TSP)-based solution, a local planner based on NBV selection, and a robot navigation module based on RL, using an occupancy grid-based SLAM algorithm. An agent that uses the SLAM output and the measurements obtained by a Lidar sensor for the navigation task was implemented by Lu *et. al.* [16]. Chen *et. al.* [17] use SLAM to compose the observation and the map reward value, indicating the map growing throughout the agent training. The slam map exploration was also explored by Botteghi *et. al.* [18], that proposes an RL and SLAM-based framework for exploring and building a map of an unknown indoor environment using the SLAM map information in the reward function and observation for the RL training. The SLAM completeness also was used by Alcalde *et. al.* [19] and Mustafa *et. al.* [20], where it is calculated at each step by obtaining the percentage of cells that are not -1 in the occupancy grid map and used for the reward function.

In addition to slam mapping, it is necessary to highlight works using deep learning models for processing high-dimensional information. Lee and Yusuf [7] used a Single Shot MultiBox Detector (SSD) MobileNet V2 model for target detection and composed an observation for the reinforcement learning agent. Yu *et. al.* [1] uses a preprocessing convolutional neural network that receives the environment information and sends this preprocessed data to the RL training, allowing changes in the environment do not require new retraining stages but only the preprocessing step. Other works that use a Deep Learning Model for processing high-level information from the environment were done by Chen *et. al.* [21] and Andriyanov [22], which uses the YoLo model to detect the objects of interest of RL training. The work of Zieliński and Markowska-Kaczmar [23] combines the vision module output, responsible for the feature extraction from an image and dimensionality reduction, with the other sensors on the robot. The integration between RL and Object Detection models can still be used for unrelated fields to robotics, such as the work of Li *et. al.* [24], which proposes a method to match traffic signals using the YoLo object detection model output with the decision-making of reinforcement learning to carry out intelligent traffic scheduling more efficiently.

Using techniques for preprocessing information from the environment allows reinforcement learning solutions to achieve better problem-solving results, making it possible to solve previously unfeasible problems only through reinforcement learning approaches. This integration uses SLAM for localization and mapping or YoLo for object detection, which extracts characteristics and high-level information with lower dimensionality from the environment.

Related works present reinforcement learning solutions that focus only on the task of exploring an unknown environment using SLAM or locating targets of interest using detection of objects that are already in the visual field of a camera. In this context, our proposal outlines an approach integrating SLAM and object detection for the task of exploring an unknown environment in search of a target. These techniques are utilized to amalgamate environmental observations, which subsequently train a reinforcement learning agent.

This agent can explore unfamiliar surroundings until it identifies a target of interest. The agent can successfully navigate and interact within a simulated environment upon observing the target.

III. Proposed Framework

In this section, we present our proposed approach for a reinforcement learning agent that can explore an unknown environment. At the beginning of the execution, the agent is challenged to explore an unknown environment. For this, SLAM mapping is used, which allows the construction of a two-dimensional occupancy map of the environment using information obtained through a simulated Lidar sensor, in addition to the robot's location. The map built by SLAM consists of a matrix of pixels, where each pixel is worth from 0 to 255, with all values reset to zero at the beginning of each training episode. During training, the agent acquires new observations, the map is updated, and the pixels representing the explored spaces are incremented, increasing gradually to the value of 255. Figure 1 shows an overview of our proposed solution.

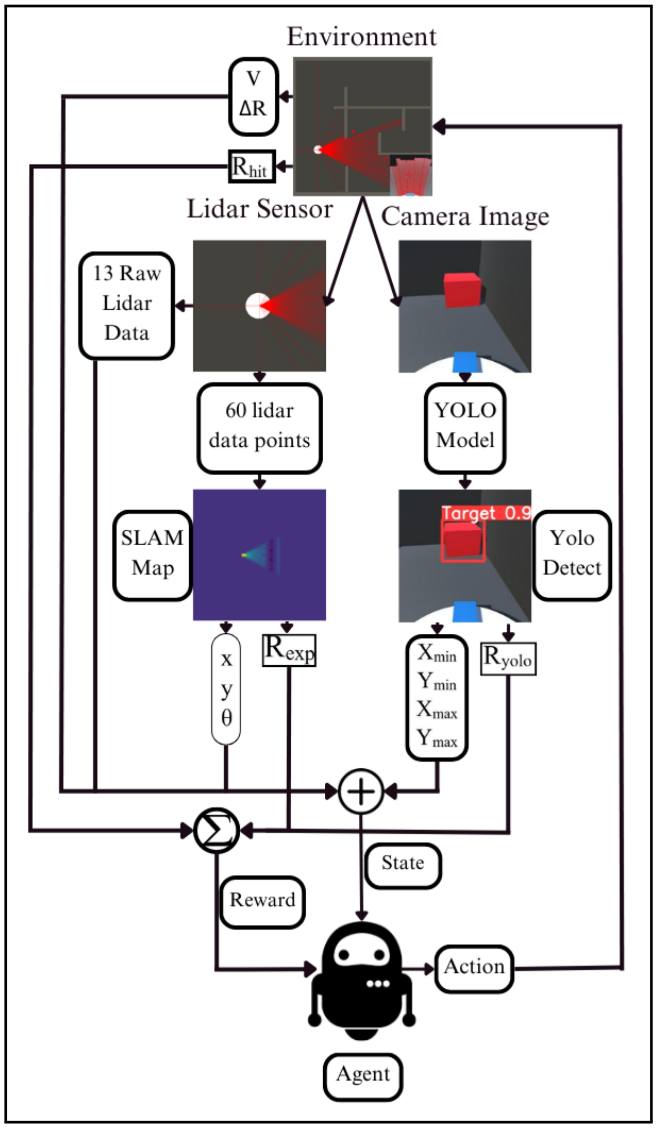


Fig. 1. Presentation of the proposal

This incremental characteristic of the map built with SLAM enables its use in monitoring the exploration of the environment, where the higher the values of the pixels, the more explored it is until it stops increasing. The incremental nature of the SLAM map was used as one of the agent's reward components during training, where at each agent training step, the weighted sum of all pixels in the map at the current instant and their difference to the map at the previous instant is performed. This increments the degree of map exploration between each map update, which is controlled by the reward R_{exp}

The mapping process allows the agent to obtain its relative position x, y, and rotation θ in the environment-mapped space without needing any extra information, but only with the mapping provided by the SLAM. This high-level and low-dimensional information will be used as the agent's state components.

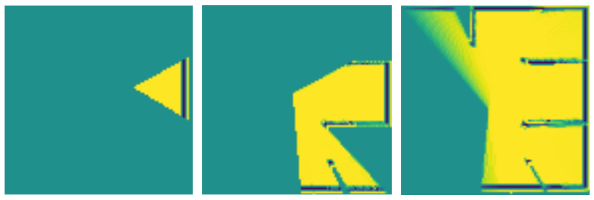
To obtain the image used during training, a camera was installed in the agent that captures images of the front of the simulated robot, which provides a visual perception of the environment immediately in front of the robot. The obtained image will then be processed by the YoLo model, which returns the bounding box (X_{min}, Y_{min}, X_{max}, Y_{max}) of each detection of objects present in the image which, for this work, will have the possibility of having only one object of interest. Therefore, just one possible detection may happen. If there is no object of interest, each component of the bounding box will have the value 0. The bounding box resulting from image processing indicates the position of the target in the image, which, given the camera's position, indicates the target's position to the agent. This was used as a component of the agent's reward R_{yolo}, where the closer the agent, the greater the reward.

In addition to the information obtained through SLAM mapping and image processing with YoLo, the agent needs to be able to perceive the environment around it and the distance from walls, doors, and obstacles. In this way, in addition to the Lidar sensor readings for mapping SLAM, this was used as the agent's raw input, with some adaptations for dimensionality reduction. For this, two Lidar sensor data were captured: frontal and backward. The work by Alcalde *et. al.* [19] proposes five frontal Lidar readings distributed at an angle of 180º in its agent, which can only walk forwards and sideways. In our case, for a better detailing of the environment, we use a set of ten readings at an angle of 120 degrees forward and three readings at an angle of 180 degrees behind, since the agent also can go backward. In addition to these inputs, the robot V's current speed and the variation of its rotation angle ΔR are sent to the agent between each step.

The primary objective of the agent training proposed in this study is to successfully reach the target following exploration of the environment and identification of the target's location. In this way, the reward obtained by the agent when performing this task is as great as possible, and the other rewards are used as partial rewards to guide the agent during training. By doing so, the agent obtains the reward R_{hit} when reaching the target.

We concatenated the outputs of the YoLo model, the SLAM mapping, the Lidar sensor, and the robot's linear velocity and angular velocity information to compose each state of the agent. Regarding the agent's reward, this is composed of the sum of the partial reward R_{exp}, R_{yolo}, and R_{hit}. Following we detail the agent's rewards composition process.

The reward component that reinforces the agent's initial exploration behavior is the R_{exp}, which is based on the difference between the exploration obtained in the SLAM map between different time instants, as can be seen in Figure 2.



(a) t_1 (b) t_2 (c) t_3

Fig. 2. SLAM Map Increment. The time instants were extrapolated for better visualization of the map growth.

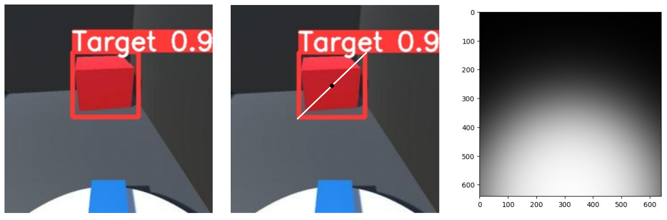
The calculation formula for finding the degree of exploitation at time t is described in Equation 1. It considers the value of each pixel about the maximum of its possible value and the map size so that the fully explored map would return the value 1. To find the R_{exp} reward value, the difference between map exploration at the current instant and the immediately previous instant during training is performed, as described in Equation 2.

$$Map_{exp}^{t} = \sum_{n=1}^{MapSize} Pixel[n]/255/MapSize \tag{1}$$

$$R_{exp} = Map_{exp}^{t} - Map_{exp}^{t-1} \tag{2}$$

Considering the stability of the created map, if the robot walks through already explored areas, new pixels with increased values will not be added to the SLAM map, and the agent's reward will be small, increasing if new pixels have their values also increased, which only happens when the robot goes through unexplored areas.

After reinforcing the exploration of the environment, the agent should be able to reinforce the target's location by the camera to hit it. For this, the output coordinates (X_{min}, Y_{min}, X_{max}, Y_{max}) of the YoLo model were used, which gave the location of the target in the image obtained by the camera, as seen in Figure 3(a)



(a) (b) (c)

Fig. 3. YoLo Detection and reward

We consider the detection center to calculate the partial reward obtained by the agent with the detection of the target, computed by tracing the diagonal between the points (X_{min}, Y_{min}) and (X_{max}, Y_{max}) and dividing it by 2, highlighted in the Figure 3(b). This point was used to calculate the partial reward R_{yolo} using its distance to the lower center point of the image, the point where the robot is located, thus stimulating the robot to reach the target. The dimension considered for the image was normalized between 0 and 1, and all calculations using this information, such as YoLo output, distances, and rewards, were performed considering this normalization. This distance was normalized using an inverse Sigmoid function, where the closer to zero is the distance, the closer to 1 will be the output of the Inverse Sigmoid, shown in equation 3. This creates a surrounding for the robot's position in the image, increasing the agent's reward, not just at the specific point where the robot is located. This is shown in Figure 3(c), with a resolution of 640x640, where the brighter area represents a greater reward for the agent, which is based on the position of the central point of object detection, being inversely proportional to the distance from the agent to the target The output value of the Inverse Sigmoid function is used for R_{yolo} reward, described in equation 4.

$$f(distance) = \frac{1}{(1 + 10^{4distance-2})} \tag{3}$$

$$R_{yolo} = f(distance).10^{-3} \tag{4}$$

The opening angle of the Lidar sensor chosen for the SLAM mapping was close to the horizontal opening angle provided by the camera. In this way, all environments mapped with SLAM would be treated and evaluated with the detection of YoLo objects, promoting a better synergy between the two solutions, in addition to allowing the agent to perceive the approach to the target using the distances obtained by the Lidar sensor. Finally, the R_{hit} reward is set to 10 when the agent reaches the goal, hits the target, and the training episode ends. Each training step was given a penalty of 0.001 to encourage the agent to obtain better scores in the shortest possible time since the longer the episode, the greater the penalty.

IV. Experimental Results and Discussion

This section will detail the achieved results through our proposal implementation For training the agent, we used the PPO algorithm [25], implemented with the Stable Baselines 3 library [12].

For the object detection task, a Yolov5n [14] pre-trained model was used, retrained with images generated from the simulated camera in the Unity environment. The images were manually annotated. For the SLAM mapping task, the BreezySLAM [13] library was used, with its default settings and a maximum map size of $12m^2$.

The environment was designed in a T format, where the agent, at the beginning of each training episode, was positioned in opposition to the target to be located in the explored environment, as shown in Figure 4(a). The map was created in such a way that the target's position was chosen randomly within some possible positions, highlighted in Figure 4(a),

Params: gamma = 0.095; learning_rate = 0.0003; batch_size = 2048; optimizer_class = torch.optim.RMSprop; other parameters were kept by default to provide a greater generalization for the agent, regardless of the target's position within the space. Additionally, targets were positioned in places where the agent had no direct vision, forcing the tasks of exploration.

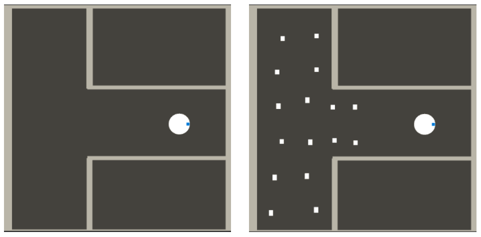

(a) Environment (b) Environment with possible target positions

Fig. 4. Environment top-view details

We conducted three training experiments, changing only the agent's reward composition. Only the R_{exp} reward was used in the first case. In the second, only the R_{yolo} reward. In the third case, the combination of the two reward components, as described in Figure 6 and 5. The three experiments intended to demonstrate the influence of each of the reward components on the agent's evolution during its training, in addition to proving its convergence when using the R_{yolo} and R_{exp} rewards together, leading the agent to reach its goal. The R_{hit} reward was kept the same in all proposed scenarios, adding the value 10 when the agent hits the object of interest.

Figure 5 indicates the reward obtained by the agent during training, where it is possible to observe the evolution of the reward of the three tested scenarios. Still, the R_{yolo} and R_{yolo} + R_{exp} configurations have similar behavior, but being the second slightly better. Scenario R_{exp}, despite presenting an improvement and reward stability, presents the worst performance.

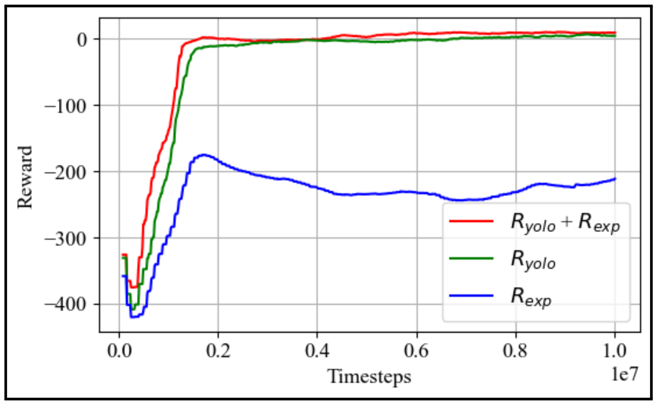


Fig. 5. Reward comparison

For a better analysis, we observed the episode duration during the training in Figure 6, which ends when the agent reaches its goal; Thus, the shorter it is, the better. In the R_{exp} scenario, the agent practically cannot reach the target throughout the training. The R_{yolo} scenario shows a strong improvement throughout the training, reaching approximately 4^3 steps duration in each episode after 10^6 training steps. However, the R_{yolo} + R_{exp} scenario was the one that presented the shortest duration, due to reaching their goal faster and having the best combination of rewards for the proposed task.

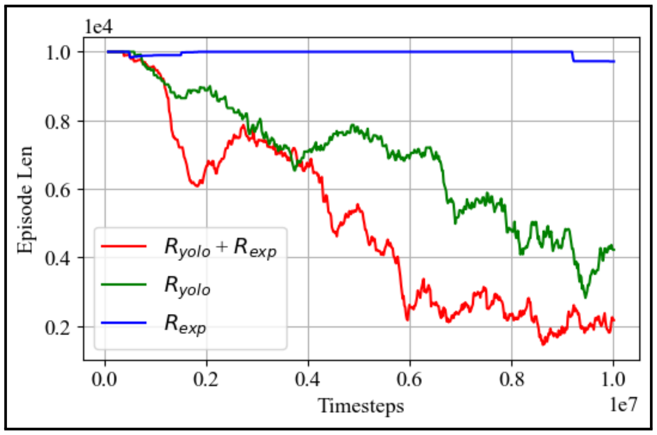


Fig. 6. Episode length comparison

After training the three agents in the scenarios tested in this work, the evaluation helper from Stable-Baselines 3 was used to evaluate the agents trained during twenty episodes. Each agent's average reward was obtained, detailed in Figure 7.

With the validation of trained agents, the difference in performance between agents is evident, with $R_{yolo} + R_{exp}$ having the best average reward.

V. Conclusion

This work presents a new approach to the reinforcement learning agent training problem to target an object of interest in an unknown environment. The possibility of pre-processing raw information makes this approach successful since the SLAM mapping is used to locate the agent in 2D space, and the YoLo object recognition model is used to locate the object of interest in the image. The modularity of the proposed solution makes it possible to change the object of interest

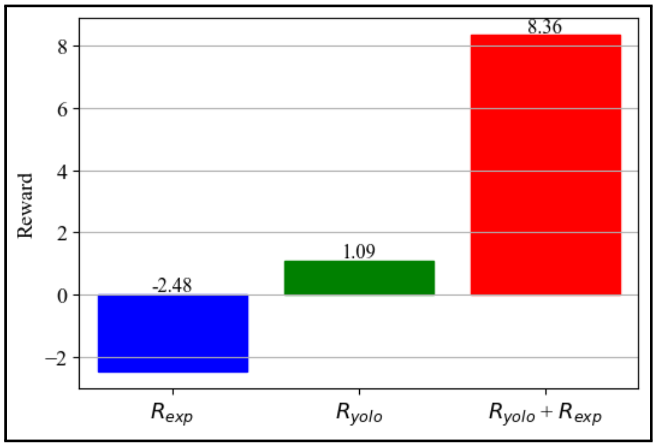


Fig. 7. Evaluation of trained Agents

without retraining the agent, just retraining the Yolo model to recognize the new object.

It is important to notice that the agent only had convergence in its training thanks to the integrated processing of raw information into high-level information and the use of these technologies for the reward composition. The combination of the technologies enabled us to achieve better results in training the agent than when using only one, as we can see in Figure 7. However, much of the success of the proposed approach can be attributed to the R_{yolo} reward, since it showed to be crucial for the agent's convergence, by providing robust learning even when used in isolation.

In addition to reward composition, the high-level information extraction process using SLAM mapping and image processing using YoLo is the main reason for the improvement and convergence of the trained agent during the construction of this work.

It is important to point out that the three agents trained in the proposed scenarios to validate the approach presented in this work obtained an improvement in the reward obtained throughout the training, being the agent that combines the R_{yolo} and R_{exp} rewards the one that obtained the best performance, therefore, it will be the combination investigated in more complex problems in future works.

The difference between this approach and the state of the art is precisely this integration between two different technologies for the same objective within reinforcement learning training. In addition to raw information processing, both were used as an instrument for composing the agent's reward, enabling its convergence in this way. Using these technologies only to process the agent's information would not allow its convergence and learning.

In future work, we intend to apply the proposed approach in more complex and larger environments and consider the impact of R_{yolo} and R_{exp} rewards in these environments and evaluate the application of curricular learning.

Acknowledgment

José Lucas Brandão Montes expresses gratitude to IFRO for granting him leave to attend the postgraduate program at Universidade Federal Fluminense.

References

[1] G. Yu, C. Zhang, J. Xu, and T. Sun, "Research on target-driven navigation of mobile robot based on deep reinforcement learning and preprocessing layer," in *Journal of Physics: Conference Series*, vol. 1575, no. 1. IOP Publishing, 2020, p. 012138.

[2] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[3] G. Yu, C. Zhang, J. Xu, and T. Sun, "Research on target-driven navigation of mobile robot based on deep reinforcement learning and preprocessing layer," in *Journal of Physics: Conference Series*, vol. 1575, no. 1. IOP Publishing, 2020, p. 012138.

[4] V. François-Lavet, P. Henderson, R. Islam, M. G. Bellemare, J. Pineau *et al.*, "An introduction to deep reinforcement learning," *Foundations and Trends® in Machine Learning*, vol. 11, no. 3-4, pp. 219–354, 2018.

[5] X. Ruan, P. Li, X. Zhu, and P. Liu, "A target-driven visual navigation method based on intrinsic motivation exploration and space topological cognition," *Scientific Reports*, vol. 12, no. 1, p. 3462, 2022.

[6] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part i," *IEEE Robotics Automation Magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[7] M.-F. R. Lee and S. H. Yusuf, "Mobile robot navigation using deep reinforcement learning," *Processes*, vol. 10, no. 12, p. 2748, 2022.

[8] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[9] Y.-L. Chen, Y.-R. Cai, and M.-Y. Cheng, "Vision-based robotic object grasping—a deep reinforcement learning approach," *Machines*, vol. 11, no. 2, p. 275, 2023.

[10] Unity, Available: https://unity.com/ Accessed: 2023-05-10.

[11] A. Juliani, V.-P. Berges, E. Teng, A. Cohen, J. Harper, C. Elion, C. Goy, Y. Gao, H. Henry, M. Mattar, and D. Lange, "Unity: A general platform for intelligent agents," *arXiv preprint arXiv:1809.02627*, 2020.

[12] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," *Journal of Machine Learning Research*, vol. 22, no. 268, pp. 1–8, 2021. [Online]. Available: http://jmlr.org/papers/v22/20-1364.html

[13] S. D. Levy, S. Bajracharya, M. Lubas, and A. Rwagaju, "Breezyslam," Feb. 2023. [Online]. Available: https://github.com/simondlevy/BreezySLAM

[14] G. Jocher, "YOLOv5 by Ultralytics," May 2020. [Online]. Available: https://github.com/ultralytics/yolov5

[15] Z. Zheng, C. Cao, and J. Pan, "A hierarchical approach for mobile robot exploration in pedestrian crowd," *IEEE Robotics and Automation Letters*, vol. 7, no. 1, pp. 175–182, 2021.

[16] C.-L. Lu, Z.-Y. Liu, J.-T. Huang, C.-I. Huang, B.-H. Wang, Y. Chen, N.-H. Wu, H.-C. Wang, L. Giarré, and P.-Y. Kuo, "Assistive navigation using deep reinforcement learning guiding robot with uwb/voice beacons and semantic feedbacks for blind and visually impaired people," *Frontiers in Robotics and AI*, vol. 8, p. 654132, 2021.

[17] S.-Y. Chen, Q.-F. He, and C.-F. Lai, "Deep reinforcement learning-based robot exploration for constructing map of unknown environment," *Information Systems Frontiers*, pp. 1–12, 2021.

[18] N. Botteghi, R. Schulte, B. Sirmacek, M. Poel, and C. Brune, "Curiosity-driven reinforcement learning agent for mapping unknown indoor environments," *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 1, pp. 129–136, 2021.

[19] M. Alcalde, M. Ferreira, P. González, F. Andrade, and G. Tejera, "Daslam: Deep active slam based on deep reinforcement learning," in *2022 Latin American Robotics Symposium (LARS), 2022 Brazilian Symposium on Robotics (SBR), and 2022 Workshop on Robotics in Education (WRE)*. IEEE, 2022, pp. 282–287.

[20] K. A. A. Mustafa, N. Botteghi, B. Sirmacek, M. Poel, and S. Stramigioli, "Towards continuous control for mobile robot navigation: A reinforcement learning and slam based approach," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLII-2/W13, pp. 857–863, 2019. [Online]. Available: https://isprs-archives.copernicus.org/articles/XLII-2-W13/857/2019/

[21] Y.-L. Chen, Y.-R. Cai, and M.-Y. Cheng, "Vision-based robotic object grasping—a deep reinforcement learning approach," *Machines*, vol. 11, no. 2, p. 275, 2023.

[22] N. Andriyanov, "Development of apple detection system and reinforcement learning for apple manipulator," *Electronics*, vol. 12, no. 3, p. 727, 2023.

[23] P. Zieliński and U. Markowska-Kaczmar, "3d robotic navigation using a vision-based deep reinforcement learning model," *Applied Soft Computing*, vol. 110, p. 107602, 2021.

[24] Y. Li, Y. Chen, S. Yuan, J. Liu, X. Zhao, Y. Yang, and Y. Liu, "Vehicle detection from road image sequences for intelligent traffic scheduling," *Computers and Electrical Engineering*, vol. 95, p. 107406, 2021.

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