International Journal of Advanced Engineering Research and Science (IJAERS)
ISSN: 2349-6495(P) | 2456-1908(O)
Vol-8, Issue-2; Feb, 2021
Journal Home Page Available: https://ijaers.com/
Journal DOI: 10.22161/ijaers
Article DOI: https://dx.doi.org/10.22161/ijaers.82.35

Robot Navigation through the Deep Q-Learning Algorithm

Madson Rodrigues Lemos[1], Anne Vitoria Rodrigues de Souza[1], Renato Souza de Lira[2], Carlos Alberto Oliveira de Freitas [1], Vandermi João da Silva[1]

[1]Federal University of Amazonas Institute of Exact Sciences and Technology, Itacoatiara, Amazonas, Brazil
[2]Cal-Comp Institute of Technology, Manaus, Amazonas, Brazil

Received: 29 Nov 2020;
Received in revised form:
25 Jan 2021;
Accepted: 14 Feb 2021;
Available online: 26 Feb 2021

***Keywords*— Deep Q-learning, robot, training, navigation**

***Abstract*— The paper aims to present the results of an assessment of adherence to the Deep Q-learning algorithm, applied to a vehicular navigation robot. The robot's job was to transport parts through an environment, for this purpose, a decision system was built based on the Deep Q-learning algorithm, with the aid of an artificial neural network that received data from the sensors as input and allowed autonomous navigation in an environment. For the experiments, the mobile robot-maintained communication via the network with other robotic components present in the environment through the MQTT protocol.*

I. INTRODUCTION

Technological advances have driven a dramatic increase in industrial productivity since the beginning of industrialization, not only to meet high demands but to guarantee the satisfaction and comfort of its consumers [1]. The great technological leaps in this sector characterize marked changes that today are called Industrial revolutions.

With the fourth industrial revolution, also called industry 4.0, smart factories are becoming a reality in many countries. They use the latest discoveries of informatics to make their factories context-sensitive, so they can deal with production turbulence in time real, using decentralized information and communication structures, for the optimized management of processes [2].

For factories to make their processes more intelligent and dynamic, the most recent and significant advances in technology that form the basis of industry 4.0 must be incorporated, among them the use of autonomous mobile robots that perform activities within production. In this scenario, according to authors Xia and Kamel [3], mobile robots must have the ability to autonomously navigate on the shop floor, needing to find a collision-free path from a starting point to a target point.

So, to make these robots independent by changing the manufacturing processes, an algorithm was applied that uses positive and negative rewards in continuous learning that reinforces the robot's experience of the submitted environment. Then, with an increase in positive rewards, the agent becomes able to locate itself through the situation in a completely autonomous way.

The algorithm that was applied is called Deep Q-learning, as it is a form of deep machine learning and relies on the use of an artificial neural network that processes the input data of the environment through sensors making it more intelligent every time. According to Faceli [4], the artificial neural network can be compared with the form of information processing carried out in the human brain that acquires experience with each new interaction with the environment through the senses found in the human body.

II. BACKGROUND

In this work, a bibliographic review was carried out to find works related to the use of reinforcement machine learning with application in mobile robots, which is the main theme of this paper. In this case, we compile 1,108 works that were collected that were related to the keywords of this article. However, after a selection that sought a more direct relationship, 15 papers were selected. Then of these, 5 researches were identified with significant results in the implementation of the method in real robots.

The authors Sasaki and colleagues [5] presented the Deep Q-network technique that derives from Deep Q-learning. The authors applied the technique to a robot that navigated in an environment using visual markings. These markings were processed by a camera allowing an intelligent agent to learn the route avoiding collision with walls. This training applied by the authors in the mobile robot, allowed to reinforce the behavior every time he got it right.

However, in this technique, the robot followed only the instructions marked on the central line, not being able to make any decision if an obstacle blocked the visual information. Another limitation of the technique in this work is the need for a robust processor to interpret the input data, as they result from the execution of a secondary convolutional neural network for image processing.

The paper of Ruan and colleagues [6] was applied a technique called Dueling Double DQN, also derived from Deep Q-learning, which uses images in depth, extracting RGB information in a Yesulated environment. However, in this experiment, the learning was based only on the processing of single visual information obtained from the images in depth. Moreover, it was carried out exclusively in a Yesulated environment, not taking into account other factors that could hinder navigation and could be detected by secondary sensors.

Saravanan and colleagues [7], it was proposed to build a mobile robot that navigated in an environment delimited by a matrix, after receiving a command via the Internet of Things. Navigation followed a system based on the classic Q-learning algorithm to reach an arbitrary object, recognizing it through a CNN (Neural Convolutional Network), and capturing it to receive the reward. Despite this work being successful in robot navigation, the environment was previously demarcated using coordinates mirrored to a fixed-size matrix, so this experiment makes it impossible for the robot to navigate in an unknown and previously unmarked environment. However, the work could contribute to the way of sending commands to the robot, using protocols for the internet of things.

According to Han and colleagues [8], the authors observed the use of a mobile navigation robot that used laser sensors autonomously using the Deep Q-learning algorithm. The robot was able to navigate in random 3D Yesulation environments. But the experiment proved to be limited due to the use of a single type of sensor. In addition, the experiment was not tested in a real environment, which could lead to the need for more input parameters for learning.

Finally, in the work of Song and colleagues [9], a Deep Q-learning technique was identified based on a memristive neural network. A model was developed to define rewards to be applied to the agent during the execution of the actions, storing the good policies on a memristive chip that has great non-volatile memory potential. However, for the application of this technique, it was necessary a specific robot that had a memristive chip for storing policies, thus making it impossible to replicate the method in common robots.

The works presented previously have very efficient results for the proposed scenarios. Still, they transform the robots into an isolated cell that does not communicate with other components present in the environment. In this work, besides the autonomous navigation mediated by Deep Q-learning with different input data, the agent was able to communicate with another robot. This robotic claw positions the parts in it after being removed from the conveyor, thus Yesulating a production environment of a factory, and despite not using any computer vision technique, the data collected by the sensors were sufficient to guide the agent in the proposed scenarios.

The comparison among the papers can be seen in TABLE I, the points taken into consideration were, autonomous navigation, use of deep reinforcement learning, use of CV (Computational Vision) techniques, experiments in a real environment, the proposed approach, and the use of communication between robots.

III. MATERIALS AND METHODS

A. *Materials used*

To build the prototypes used in the experiment, the educational robotics kit Lego Mindstorms in the NXT and EV3 versions were used, which made it possible to build a navigation robot represented by the car 1, a manipulator (robotic arm) 2 and a treadmill 3, according to Fig.1.

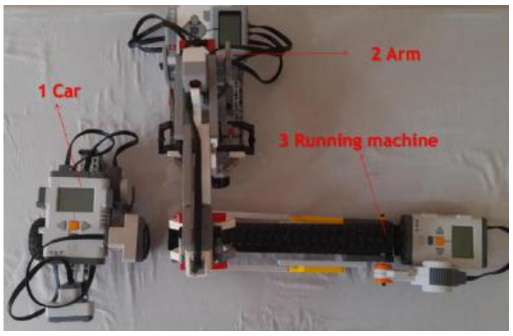


Fig. 1. Robotic prototypes used in the experiments

Table.1: Comparison between the papers and this works

Paper	Navigation Autonomous	DQN	CV	Real environment	Approach	Agents Communication
Sasaki (2017)	Yes	Yes	Yes	Yes	Navigation through visual markings identified by a camera and processed by a secondary CNN.	No
Ruan (2019)	Yes	Yes	Yes	No	Navigation through in-depth images and analysis of RGB information in a Simulated environment.	No
Saravanan (2019)	Yes	No	Yes	Yes	Navigation by mirrored coordinates in a fixed-size matrix processed by classical Q-learning.	No
Han (2018)	Yes	Yes	No	No	Autonomous navigation in 3D simulation based on laser sensors as an entry to Deep Q-learning.	No
Song (2018)	Yes	Yes	No	Yes	Learning based on a memristive neural network, with a model for defining rewards and storing good policies on a memristive chip.	No
This paper	Yes	Yes	No	Yes	Learning using Deep Q-learning for autonomous navigation based on ultrasonic and RGB sensors as input to neural network, being viable for applications in common robots. Navigation is only started after communication between agents.	Yes

B. *System Architecture*

The system architecture is composed by the blocks that represent the integral parts of the intelligent system, indicating its components, forms of communication, and behaviors, as shown in Fig. 2.

The mobile robot is responsible for transporting the parts that leave a starting point represented by the production line and take it to a target point designated by the dispatch of the factory. So that it could navigate to its target point, the Deep Q-learning algorithm was used, which makes the agent more efficient according to his experience on the environment, enabling autonomous navigation, choosing a collision-free path.

The communication between the two robots took place through a local wireless network that allows the message to be forwarded through the MQTT (Message Queue Telemetry Transport) protocol that is specific for M2M (machine to machine) communication and for the internet of things. The message reaches an MQTT broker that plays the role of FOG, in analogy to a local Cloud present on the intranet, being accessed regularly by the robots. As the LEGO NXT robot only allows connections via Bluetooth networks, a data exchange feature was used between the robot and the server via the Bluetooth socket. This procedure allows a communication bridge between the slave robot with the broker and the master robot that makes direct access via the Wi-Fi network.

When establishing a connection with a broker, robots can perform actions to publish to write a message and subscribe to read the message received.

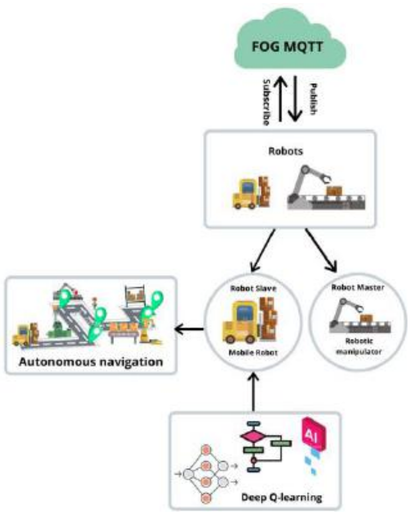


Fig. 2 Experiment architecture

C. Deep Q-learning algorithm

The formal definition of reinforcement learning is given by the Bellman equation, where:

- ***V(s)*** : Present value of a state ***s***;
- ***R (s,a)***: Reward related to action a in state ***s***;
- ***V(s')***: Future value in future state ***s'***;
- ***a:*** Action taken by the agent;
- ***s:*** Current agent state
- ***γ:*** Discount factor.

The discount factor γ notifies the agent when he is close to his destination, according to (1) [10].

$$V(s) = max_a\big(R(s,a) + \gamma V(s')\big) \qquad (1)$$

For the Bellman equation to be applied to Q-learning, the formula undergoes some modifications so that it can calculate the quality of the actions for each agent state in current time (t) and in an earlier time (t-1), according to (2).

From the Q-learning equation (2), the construction of the Deep Q-learning algorithm was started, which makes Q values available according to the agent's state, so that actions can be taken.

To calculate the Q values in the state of the agent, a dense artificial neural network was used with four input. The values were captured by sensors, a hidden layer with 30 artificial neurons that perform the Q-learning calculation, as output from the network four values are presented for Q, as shown in the network diagram in Fig. 3.

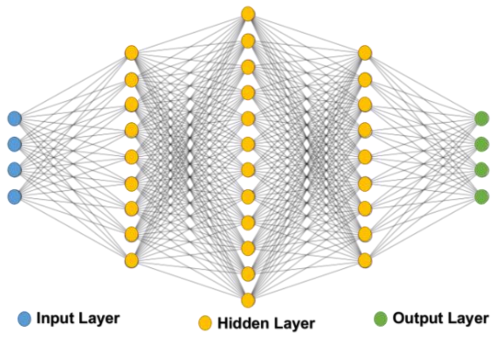
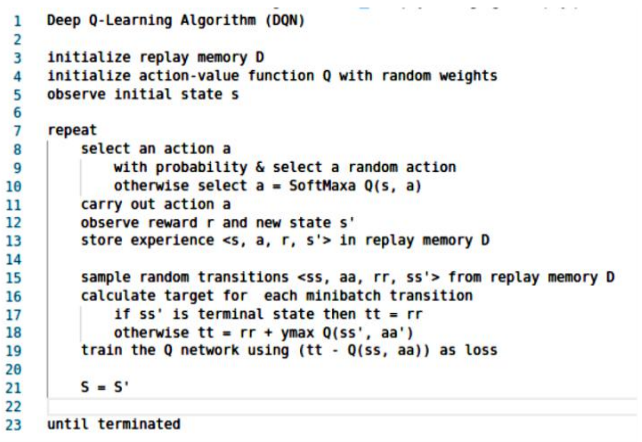


Fig. 3. Neural network used in the learning algorithm.

For the final action of the agent to be taken, a mathematical equation called SoftMax, see in equation (3), was used, which calculated which response was most likely to be chosen based on the Q weights processed by the network. After making a decision, when the agent made a good decision, he received a positive reward, when the opposite occurred, he received a negative reward.

$$SoftMax = \frac{e_Q}{\sum e_Q} \qquad (3)$$

Deep Q-learning becomes more and more efficient with each interaction. To guarantee this statement, a technique called replay experience was used, acting on the agent to remember the last decisions made in the state he is in. Hence, he sends these values again for the entrance of the network, being able to maintain the same decision if he was positively rewarded in the last interaction or choose a different action if he was punished.



```
Deep Q-Learning Algorithm (DQN)

initialize replay memory D
initialize action-value function Q with random weights
observe initial state s

repeat
    select an action a
        with probability & select a random action
        otherwise select a = SoftMaxa Q(s, a)
    carry out action a
    observe reward r and new state s'
    store experience <s, a, r, s'> in replay memory D

    sample random transitions <ss, aa, rr, ss'> from replay memory D
    calculate target for  each minibatch transition
        if ss' is terminal state then tt = rr
        otherwise tt = rr + ymax Q(ss', aa')
    train the Q network using (tt - Q(ss, aa)) as loss

    S = S'

until terminated
```

Fig.4. Deep Q-learning algorithm

Source: Adapted from AILEPHANT

The algorithm can be implemented in any programming language and applied to a virtual or real agent. For this work, the algorithm was written in the python programming language, as it allowed greater flexibility in the code syntax and the access of the robot resources. The pseudo-code in Fig. 4 shows the basic writing of the Deep Q-learning algorithm used in the experiment.

IV. EXPERIMENTS AND RESULTS

For this work, two scenarios were developed, one free of obstacles and the other with obstacles.

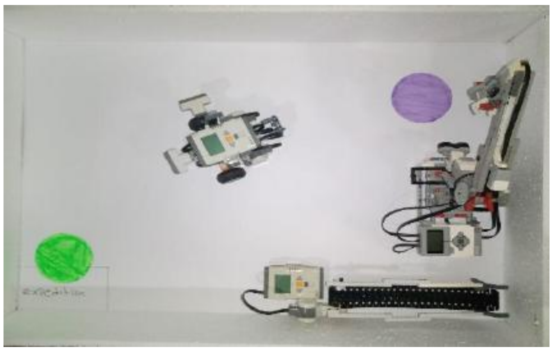


Fig. 5. Navigation Scenario Template one

In this scenario, the robot could navigate freely through the environment only having to complete its objective in the shortest time. Without approaching the walls of the scenario, the rewards received +0.01 if the agent reached his goal, if he touched the wall his sensors a bad action, receiving a penalty of approximately -0.01 and increased if the agent remained in error. The rewards were recalculated for each new state and action pair represented by R (s, a).

Learning success was measured using the graph of the score of rewards received by the agent, given by the average rewards, and the score values were generated for every 1000 rewards generated (4).

$$Score = \frac{\sum_i^n (R(s,a))}{n} \quad (4)$$

The rewards started with negative values in the first interactions due to the punishments received, as the agent tried to reach his goal by sailing very close to the walls and started to collide several times. However, as the number of interactions increased, the agent began to navigate a shorter and collision-free path, thus receiving positive rewards. At the end of the test, the agent found a diagonal way that took him more quickly to his target with a hit rate of 91.2% obtained by the percentage of the score of good deeds concerning the total number of interactions and neglecting bad deeds, with this, the learning success was verified, which can be seen by the approximation of the linearity of the positive rewards of the agent marked by the green line, as shown in the graph of Fig. 6.

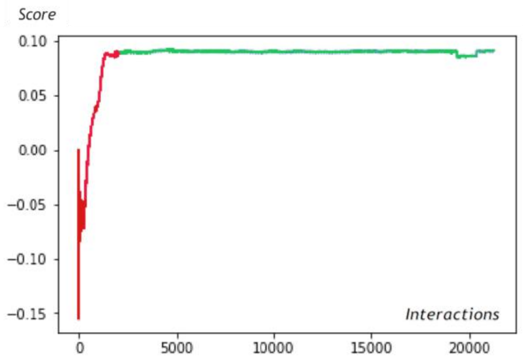


Fig. 6. Graph of agent rewards in scenario one

The second experiment was carried out in a scenario with a higher degree of difficulty, where the agent started from the starting point marked by a purple circle and should reach the target point with the green mark. However, in this scenario the agent could only start navigation from the command sent by the robotic manipulator, indicating that the parts could already be transported. When starting the navigation, the robot was able to use two navigation options, followed a mark on the ground using its RGB sensor that successively shines red, green and blue light on the way, the reflected light is collected by a light sensor sensitive to the entire length wave, differentiating the dark trail from the rest of the environment, indicating the path to be taken, however, losing track, managed to verify through its ultrasonic sensors possible obstacles present in the route, these sensors were able to detect an object and measure its proximity in distances of less than 10 centimeters, avoiding the obstacle and maintaining navigation until finding the mark again or reaching the target point. The model for this scenario can be seen in Fig.7.

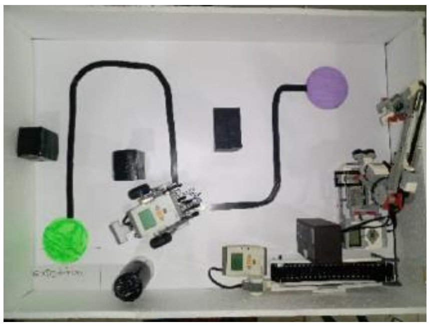


Fig. 7. Navigation Scenario 2 Template

To measure the quality of learning in the second scenario, policies similar to those used in the first scenario were used but adding reward for the agent's proximity to his goal and additional punishments if the agent collided with other components of the environment or lost his track.

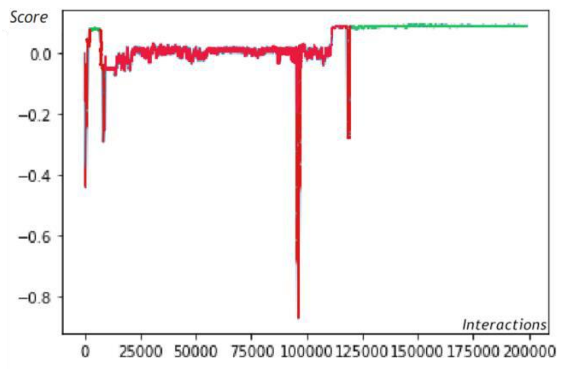


Fig. 8. Graph of agent rewards in scenario 2

In Fig. 8, the agent's rewards graph for the second scenario presented a large number of variations, mixing the learning between mistakes and successes in the initial interactions due to the number of punishments received when colliding with other elements of the environment, losing their track or crash into the wall. However, after approximately 125,000 interactions, the agent started to navigate in an expected way, following the marked and collision-free path, receiving the positive reward and maintaining its linearity of positive actions marked in green, indicating the good adherence of the algorithm to the robot. Navigation as an 87.5% hit rate, very good considering the complexity of the test environment, in addition to other factors that influenced learning in the real scenario, for example, the response time of sensors when obstacles were already in a very short distance from the agent.

V. CONCLUSION

Therefore, through experiments, we arrived at an intelligence model that can be applied as part of the automated production process in factories, as it was possible to transport parts from one place to another using an autonomous mobile robot. It was also verified the efficiency of the communication system between the robots through Fog MQTT, as it presented a practically instantaneous time in sending and receiving messages. Thus, it is expected that this work will contribute to the incorporation of Deep Q-learning in robots. Collaborative activities that carry out activities within real factories, strengthening the concepts of industry 4.0.

As future work, improvements will be made to the algorithm to perform more complex tests with dynamic environments. It is also intended to apply Deep Q-learning to a real industrial robot to prove the feasibility of its use in a real factory, as the research was limited to the use of educational robots.

ACKNOWLEDGEMENTS

This work was developed with support from the Calcomp Electronic by the R&D project in the Institute of Exact Sciences and Technology of the Federal University of Amazonas in Itacoatiara – AM. (Contract No. 001/2019), and the FAPEAM institution through the research grant of the PAIC UFAM Program 2019-2020 is also registered.

REFERENCES

[1] Rüßmann, M., Lorenz, M., Gerbert, P., Waldner, M., Justus, J., Engel, P., & Harnisch, M. (2015). Industry 4.0: The future of productivity and growth in manufacturing industries. Boston Consulting Group, 9(1), 54-89.

[2] Lucke, D., Constantinescu, C., & Westkämper, E. (2008). Smart factory-a step towards the next generation of manufacturing. In Manufacturing systems and technologies for the new frontier (pp. 115-118). Springer, London.

[3] Chen, X. I. A., & El Kamel, A. (2015). A reinforcement learning method of obstacle avoidance for industrial mobile vehicles in unknown environments using neural network. In Proceedings of the 21st International Conference on Industrial Engineering and Engineering Management 2014 (pp. 671-675). Atlantis Press, Paris.

[4] Faceli, K., Lorena, A. C., Gama, J., & Carvalho, A. C. P. L. F. (2011). Inteligência Artificial: Uma abordagem de aprendizado de máquina. Rio de Janeiro: LTC, 2, 192.

[5] Sasaki, H., Horiuchi, T., & Kato, S. (2017, September). A study on vision-based mobile robot learning by deep Q-network. In 2017 56th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE) (pp. 799-804). IEEE.

[6] Ruan, X., Ren, D., Zhu, X., & Huang, J. (2019, June). Mobile robot navigation based on deep reinforcement learning. In 2019 Chinese control and decision conference (CCDC) (pp. 6174-6178). IEEE.

[7] Saravanan, M., Kumar, P. S., & Sharma, A. (2019, July). Iot enabled indoor autonomous mobile robot using cnn and q-learning. In 2019 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT) (pp. 7-13). IEEE.

[8] Han, S. H., Choi, H. J., Benz, P., & Loaiciga, J. (2018, January). Sensor-based mobile robot navigation via deep reinforcement learning. In 2018 IEEE International Conference on Big Data and Smart Computing (BigComp) (pp. 147-154). IEEE.

[9] Song, W., Zhou, Y., Hu, X., Duan, S., & Lai, H. (2018, August). Memristive Neural Network Based Reinforcement Learning with Reward Shaping for Path Finding. In 2018 5th International Conference on Information, Cybernetics, and Computational Social Systems (ICCSS) (pp. 200-205). IEEE.

[10] Bellman, R. (1954). The theory of dynamic programming. Rand corp santa monica ca.