Int. J. Advanced Design and Manufacturing Technology, 2024, Vol. 17, No. 3, pp. 39-48

DOI: 10.30486/ADMT.2024.1126477 *ISSN: 2252-0406* *https://admt.isfahan.iau.ir*

Determination of Reinforcement Learning Reward Parameters to Solve Path Planning of Unknown Environments by Design of Experiments

Issa Alali Alfares, Ahmad Reza Khoogar *
Faculty of Materials & Manufacturing Technologies, Malek Ashtar University of Technology, Tehran, Iran
E-mail: issa.alifares@mut.ac.ir, khoogar@gmail.com
*Corresponding author

Received: 14 July 2024, Revised: 9 August 2024, Accepted: 8 September 2024

Abstract: The Reinforcement Learning Approach (RL) is used to solve the path-planning problem of an autonomous mobile robot in unknown environments. Despite that RL is a recent and powerful tool, it requires a lot of training processes because there are so many parameters in the agent's training process. Some of these parameters have a larger effect on the convergence of the learning process than others, so, knowing these parameters and their suitable values makes the training process more efficient, saves time, and consequently makes the trained agent execute the required task successfully. No analytical equations are available to determine the best values for these parameters, therefore, in this paper, a statistical analysis is made using the design and analysis of experiment (DoE) methods to determine the parameters that have the largest effect on the training process. After that, analysis is done to determine the values of the most effective parameters. Results show that the determined parameters lead to a successful autonomous path planning in different unknown environments

Keywords: Autonomous Path Planning, Design of Experiment, Mobile Robot, Reinforcement Learning, Reward, Training Parameters

Biographical notes: Issa Alali Alfares received his MSc in applied design and mechatronics from Malek Ashtar University of Technology (MUT), Iran, in 2010. Currently, he is a PhD student at Malek Ashtar University of Technology Tehran, Iran doing his research in robot navigation using machine learning and artificial intelligence methods. **Ahmad Reza Khoogar** received his PhD in Mechanical Engineering from the University of Alabama in Tuscaloosa in 1990. He is currently an Associate Professor at the Faculty of Materials & Manufacturing Technologies, Malek Ashtar University of Technology Tehran, Iran. His current research interests include Robotics and Mechatronic systems.

Research paper

1 INTRODUCTION

Mobile robot applications have been spread almost in everyday human life, and robots must be able to move efficiently and safely in various environments. Path planning is the fundamental task necessitated by any mobile robot and it is the most important part of any navigation process, it must address challenges such as perception, cognition, human-robot interaction, and obstacle avoidance. There exists a large number of path-planning methods that have been developed in recent decades to ensure the generation of collision-free trajectories. However, each method has its own set of advantages and disadvantages, thereby, this field of research is still one of the most extensively studied and investigated scientific domains.

Path planning methodologies are divided into two primary categories: classical and intelligent [1]. The major drawbacks of the classical approaches are the computational expenses, and the inability to deal with unknown environments [2]. Most of the new studies focus on intelligent methods such as artificial neural networks [3], reinforcement learning [4], genetic algorithms [5], fuzzy logic [6], cuckoo search [7], and various other techniques which are mainly based on artificial intelligence [8], and the ability of the robot to learn enabling it to acquire knowledge and make decisions in new scenarios.

There is a tendency among researchers to solve the mobile robot navigation problem using reinforcement learning (RL) methods since they are simple, diverse, and more suitable for unknown systems that have a great amount of uncertainty. The main problem with RL methods is that they require a lot of training processes to reach a successful training agent. The classical method is to try different values of these parameters manually but it is a very time-consuming operation.

Design and Analysis of Experiment (DoE) is a statistical method widely used for product design and development as well as optimization [9-10]. These methods find their way into almost all fields of engineering. For instance, in mechanical engineering, DoE methods are applied to study the suspension system of a car to improve the ride comfort of a vehicle at various speed domains [11], whereas Hussain et al. employ DoE methods to find the best combination of the parameters of the device that is used to measure the cylindricity of the engine cylinder core [12]. Another field for applying DoE methods is civil engineering and road construction in [13]. Also, DoE methods are applied vastly in material sciences and the study of the corrosion process on a steel alloy in a harsh salty environment [14]. Moreover, due to the simplicity and time cost-saving of DoE approaches, they are applied in medical researchers in radiotherapy [15]. Unlimited list can be mentioned here, in transport [16], product reliability improvement [17], electronics, and computer products industry [18], Energy production [19] and many other fields.

Other approaches are used in conjunction with DoE methods to execute efficient research. Vieira et al [20] employ the design and analysis of experiments approach in combination with an artificial neural network (ANN) to improve the operation of a steam generator used for a power generation plant. A model to estimate the efficiency of the steam generator using ANN is built then the DoE method is applied to study the effect of seven parameters on the steam generator performance. While another study applied the DoE with ANN to evaluate and optimize a chemical material that is used to produce fuels and other chemical products [21]. So, it can be said that DoE is a powerful and reliable tool to study the effect of multiple parameters on a certain process. DoE gives the best inference possible of the executed experiments and it notably decreases the total number of the necessary experiments.

In this paper, DoE is used to determine the parameters of a new reward function of a reinforcement learning agent defined especially to solve the path planning of an autonomous mobile robot in different unknown environments. The main goal of using DoE is to reduce the total number of the training operations of the RL agent which takes a very long time to be done.

The remainder of the paper is organized as follows: In the following section, a theoretical background is stated to clarify the definition of the new proposed reward function then a factorial design is suggested to decrease the total number of training operations and all the data collected is summarized in the designed table of experiment. After that analysis of the data to determine the parameters with the highest effect on the training process is done. To evaluate the training agent with the obtained values, experimental tests are carried out to solve path planning in different environments. Finally, a summary of what has been achieved in this paper is presented in the conclusion section.

2 THEORITICAL BACKGROUND

Reinforcement learning (RL) approaches depend on the interaction between the agent and the environment to achieve the learning of the agent. No previous model is required for RL methods, because they depend on trial and error to learn the agent. To solve any problem with RL, three phases must be carried out. Setup phase, training phase and test phase. In the setup phase, the problem should be defined and formulated which means to choose a suitable and meaningful definition of the RL problem components; the state space, the action space, and the reward signal.

The most distinguishing feature of RL is that taking good actions after training depends on maximizing a reward function which is the most important component of the RL method. In order to define the reward function, previous studies and research divide the state space into four or five kinds of states: safe, unsafe, winning, and failure state [22-25]. Then they define the reward signal as a real number for each transition from one kind of state to another kind of state. Other researchers used a reward function that assigns a value of (+1) to the goal state, a value of (-1) for the collision state, and a value of (0) to all other states [26]. Other studies [27] even consider a much simpler definition of the reward to be (-1) if the distance between the robot and the obstacles becomes less than a certain limit, and a value of (0) otherwise. All the previous definitions simplify the real environment by considering an infinite number of real situations as one state, but this representation is unrealistic and inaccurate; however, those methods can find solutions only in simple cases and they are not applicable in complicated real environments.

To tackle this problem, we propose a totally new definition of the reward function taking into consideration the following points:

- The agent should receive a high positive reward for successfully reaching the target position because this behavior is highly desired.
- The agent should be highly penalized if it collides with an obstacle.
- The agent should get a small positive or a small negative reward due to the transition from one state to a "better" or "worse" state.

So, we suggest a total reward function r which is a weighted sum of five partial rewards r_1, r_2, r_3, r_4, r_5. These partial rewards are related respectively to: the distance between the robot and the goal point, the changes in distance between the robot and the goal, the distances between the robot and the different obstacles, the changes in distances between the robot and the obstacles, and finally how long the robot takes to reach the goal.

$$r = a_1.r_1 + a_2.r_2 + a_3.r_3 + a_4.r_4 + a_5.r_5 \tag{1}$$

Where; a_1, a_2, a_3, a_4, a_5 are the weights of partial rewards and the value of these parameters will be determined during the training process of the RL agent which is represented by a deep neural network. These weights express the relative importance of the different parts of the total reward, and determining the best values of these parameters is the core of our study.

As for r_1, it is related to the distance between the robot and the target. This amount must take an increasingly high value as the robot gets close to the target. So, r_1 is suggested to be in the next form:

$$r_1 = \frac{1}{(d_t(r,T))^2} \tag{2}$$

Where: $d_t(r,T)$ is the distance between the robot and the goal point at step t. "Eq. (2)" shows that if the distance between the robot and the target $d_t(r,T)$ becomes smaller, the reward r_1 will be bigger.

As for r_2, it takes the highest value possible of the reward, which is $(+100)$ when the robot reaches its target. Otherwise, the value of r_2 will depend on the distance change between the robot and the target. If the robot is coming closer to the target, then it will receive a positive reward proportional to the rate of how much this closeness happens. Symmetrically, if the robot is going farther from the target, then it will receive a negative reward proportional to the rate of how much this change happens.

$$r2 = \begin{cases} +100 & \mathit{if\ the\ robot\ reaches\ the\ goal} \\ \dfrac{d_{t-1}(r,T) - d_t(r,T)}{max(d_{t-1}(r,T), d_t(r,T))} & \mathit{otherwise} \end{cases} \tag{3}$$

Where: $d_{t-1}(r,T)$ is the distance between the robot and the goal at step $t-1$.

Regarding r_3, it is related to the distance between the robot and the different obstacles. Of course, if the distance between the robot and the obstacles becomes smaller, the negative reward must be bigger in this case. Taking that into consideration, we suggest the following definition of r_3:

$$r_{3i} = \frac{-1}{(d_t(r,o_i))^2} \qquad i = 1,2,\dots,n \tag{4}$$

$$r_3 = r_{31} + r_{32} + \cdots + r_{3n} \tag{5}$$

Where; $d_t(r,o_i)$ is the distance between the robot position and the obstacle position detected in direction i at step t. n is the number of the directions that the robot senses the existence of obstacles in that direction which falls in the range of the sensor.

As for r_4, It depends on the distance changes between the robot and other obstacles that are moving and sensed by the robot in different directions. We suggest the following definition of r_4:

$$r_{4i} = \frac{d_t(r,o_i) - d_{t-1}(r,o_i)}{d_t(r,o_i)} \qquad i = 1,2,\dots,n \tag{6}$$

$$r_4 = r_{41} + r_{42} + \cdots + r_{4n} \tag{7}$$

Where; $d_{t-1}(r,o_i)$ is the distance between the robot position and the obstacle position detected in direction i at step $t-1$.

As for the last part of the reward r_5, it is just a constant reward that reflects the effect of the number of steps needed to reach the goal, or in other words the less time spent to reach the goal the better performance the robot makes. So, a negative reward(punishment) is added at each extra time step. So, as stated above the main goal of this study is to find the best values of the parameters a_1, a_2, a_3, a_4, a_5 defined in “Eq. (1)”.

3 PREPARING AND GATHERING DATA

In this section, the second phase of the RL approach which is training the RL agent and gathering the data needed to determine the best values of the reward function is explained.

In the RL approach, the training is done through the interaction between the agent and the environment. The agent that we propose to learn is a deep Q-neural network (DQN). To do that, a MATLAB platform was built to represent the simulation environment in which the RL algorithm will be implemented. This platform is provided with all the information that may be needed in the simulation process like the environment where the navigation is taking place, the start point, the goal point, and all the information related to the obstacles like their numbers, shapes, and positions, movement, and their speed (“Fig. 1”).

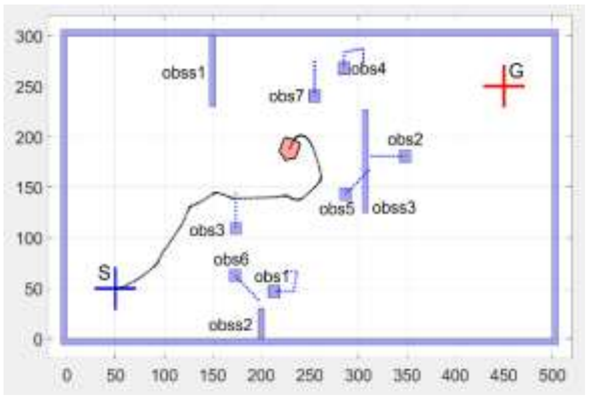

Fig. 1 The geometrical environment proposed for the training. S: start position, G: goal position, obss1, obss2, obss3: static obstacles. obs1, obs2,…, ob7: dynamic obstacles.

Also, this platform contains the definition of the state space, action space, and reward function. Finally, the platform enables the animation of the robot's motion, obstacles motion, and the development of the training process. So, one of the main goals of this platform is to generate the data needed for the training process and to test the trained robot. Figure 1 shows the proposed simulation environment for training which contains three static obstacles and seven moving ones.

The classical method to determine good values of the training parameters is to try a random group of values until we accidentally reach acceptable results. RL has too many parameters to adjust during the training process like: the learning rate for the neural network, number of layers, number of neurons in each hidden layer, mini-batch size, regularization parameter, dropout parameter, replay memory size, discount rate γ, number of maximum episodes, number of maximum steps per episode, and many other parameters and with our suggested definition of the reward, other parameters have been added so this will necessities a large number of training processes so we suggest to design an experiment from the beginning in order to save time.

Knowing that every single complete training process takes between 12 to 30 continuous hours; the 2^k factorial design came to mind. The goal of this design is to determine the most effective parameters of the reward function defined in “Eq. (1)”. But first, a rough training operation is conducted to determine a rough domain for each value of the parameters a_i at which the training process has been converged. The next domain values are found:

$a_1 \in [0.01,0.5]$, $a_2 \in [1,20]$, $a_3 \in [0.01,0.5]$, $a_4 \in [1,2]$, $a_5 \in [0.1,2]$.

In the definition of our reward function, there are 5 parameters to determine and each parameter has 2 values then, the total number of experiments needed is $2^5 = 32$. Therefore, a table that includes all possible combinations of the reward parameters is prepared (Table 1). After each training process, a test process is executed. The test environment is different from the training environment. Figure 2 shows the test environment used to evaluate the trained agent.

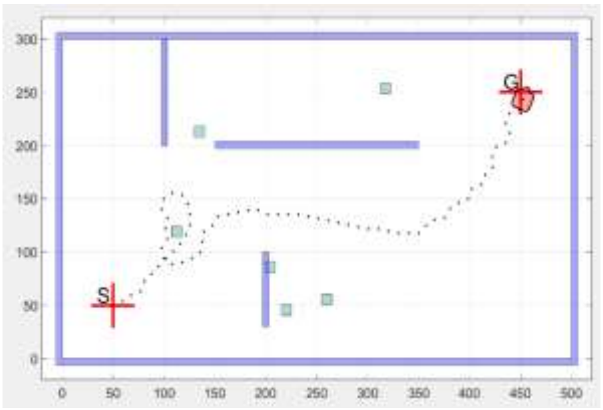

Fig. 2 The test environment proposed for evaluating the success rate after each training process.

This environment contains three static obstacles and six moving ones. Dimensions and positions of obstacles in the test environment are different from those in the training environment. The trained agent is tested and the test is repeated three independent times. Each time, the robot has to conduct the test one thousand times, and in the end, the percentage of the robot's success in reaching the target, which is called the "Success rate" is calculated. The results are shown in "Table 1". The last three columns contain the values of the success rate for each experiment conducted.

Table 1 The success rate obtained for all the combinations of the reward parameters

trial	a_1	a_2	a_3	a_4	a_5	Success rate1 %	Success rate2 %	Success rate3 %
1	0.01	1	0.01	1	0.1	45.1	44	43.8
2	0.5	1	0.01	1	0.1	43.8	45	44.4
3	0.01	20	0.01	1	0.1	45	44.9	45
4	0.5	20	0.01	1	0.1	44.8	43.5	45.4
5	0.01	1	0.5	1	0.1	55.7	56.9	55.8
6	0.5	1	0.5	1	0.1	57.1	57.2	56.5
7	0.01	20	0.5	1	0.1	56.4	57.5	56.3
8	0.5	20	0.5	1	0.1	56.9	57	55.7
9	0.01	1	0.01	2	0.1	28.3	28.4	27.6
10	0.5	1	0.01	2	0.1	28.2	29	28.3
11	0.01	20	0.01	2	0.1	29.3	29.4	29.6
12	0.5	20	0.01	2	0.1	29.3	29.9	29.2
13	0.01	1	0.5	2	0.1	39.8	40.4	39.2
14	0.5	1	0.5	2	0.1	39.4	40.3	40.1
15	0.01	20	0.5	2	0.1	40.2	40.5	40.9
16	0.5	20	0.5	2	0.1	40.6	40.8	40.6
17	0.01	1	0.01	1	2	82.5	82.8	82.6
18	0.5	1	0.01	1	2	82.4	82.5	82.8
19	0.01	20	0.01	1	2	82.8	82	82.7
20	0.5	20	0.01	1	2	82.2	83	83.1
21	0.01	1	0.5	1	2	94.4	95.2	94.6
22	0.5	1	0.5	1	2	94	93.8	95.2
23	0.01	20	0.5	1	2	94.8	94.9	95.2
24	0.5	20	0.5	1	2	94	95.1	95.3
25	0.01	1	0.01	2	2	65.3	66.2	66
26	0.5	1	0.01	2	2	66	66.1	65.5
27	0.01	20	0.01	2	2	65.4	64.9	65.3
28	0.5	20	0.01	2	2	66	66	65.2
29	0.01	1	0.5	2	2	78.1	78.7	77.8
30	0.5	1	0.5	2	2	78.3	78.5	77.6
31	0.01	20	0.5	2	2	78.2	78.4	78.3
32	0.5	20	0.5	2	2	78.1	78.5	77.9

4 DETERMINATIONS OF REWARD PARAMETERS BY DESIGN OF EXPERIMENT

After the data was prepared, analysis was done with Minitab application and the following results were obtained. First, the normal distribution of effects ("Fig. 1") shows the effective factors on the success rate. The factors that are far from the line that represents the normal distribution are the most effective factors. Generally, the farther the factor from the line, the bigger the effect that it has. So, the most effective coefficients on the success rate are as follows, starting with the most effective ones:

1. Coefficient related to the distance change to obstacles, a_4.
2. Coefficient related to the distance to obstacles, a_3.
3. Coefficient related to the distance to the target, a_1.

Validation of the previous results is done through the observation of the residuals graphs ("Fig. 3") from which we can depict the following:
Figures 4a and b show the normal probability plot and the histogram of the residuals. It shows clearly that the residuals follow a Gaussian distribution. In "Fig. 4c and d", no particular pattern is followed by the residuals and it can be stated that these values show randomness, which assures the validation of the obtained results.
The effect of the five coefficients on the success rate is depicted independently in "Fig. 5 and Fig. 6". It shows that the coefficients related to the distance change of the target a_2 and the number of steps a_5 have negligible effect on the success rate. Whereas the remaining coefficients a_1, a_3, a_4 have a remarkable effect on the success rate as depicted previously in "Fig. 3".

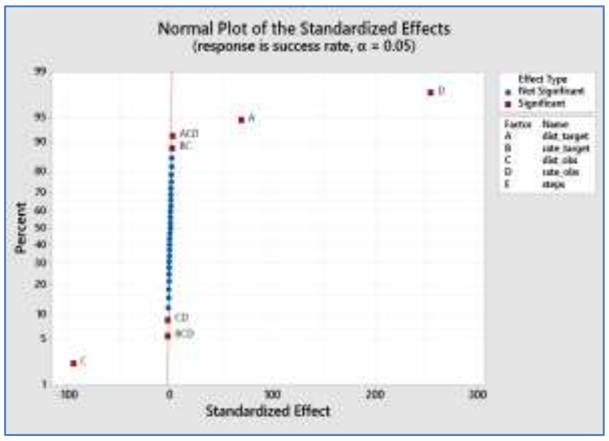

Fig. 3 Normal plot of the effects of all the reward coefficients and their interactions.

Since a_2, a_5 are not very effective, their values have been taken in the middle of their ranges. Figures 5 and 6 show an increasing success rate with increasing values of the parameters a_1, a_4 and with decreasing values of a_3 , so, a_1, a_4 have been taken to their highest limits whereas a_3 has been considered in its lower limit. The following values of all the parameters, that achieve a high success rate which is 98.2% are considered:

$a_1 = 0.5$, $a_2 = 10$, $a_3 = 0.01$, $a_4 = 2$, $a_5 = 1$

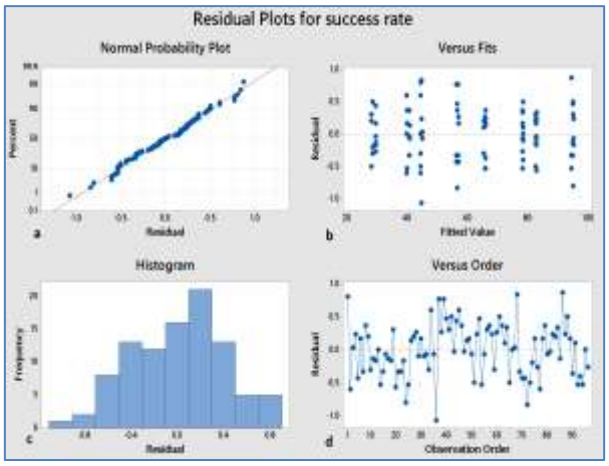

Fig. 4 The residuals graphs of the data.

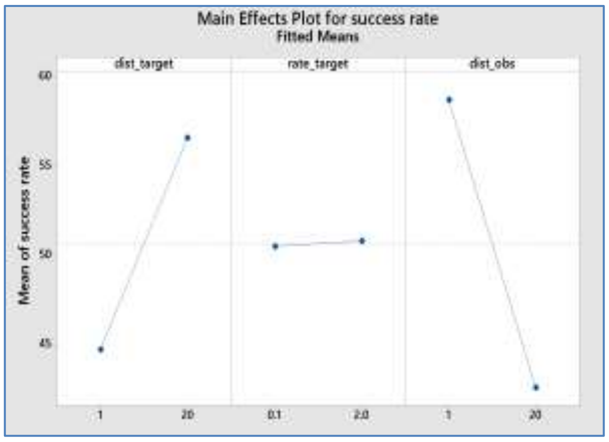

Fig. 5 The effects of a_1, a_2, a_3 on the success rate.

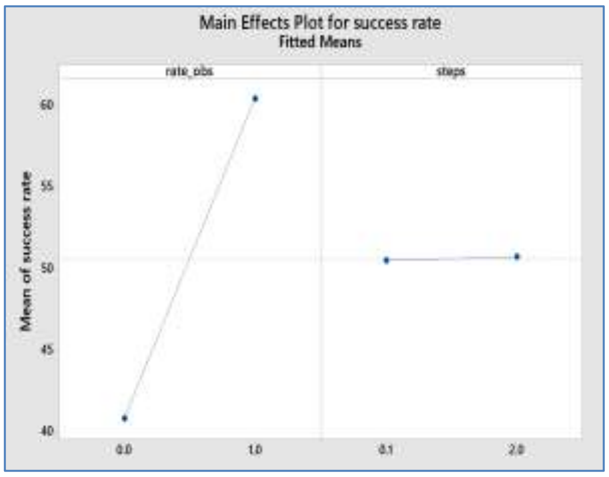

Fig. 6 The effects of a_4, a_5 on the success rate.

5 PATH PLANNING RESULTS

This section contains the tests of the trained agent with the resulting values of reward parameters. This test is done in totally new environments. Simulation tests and also real tests on real robots are done.

The environment consists of a group of obstacles like barriers forming a kind of maze between the start and the goal points, and then a few adjustments are made to get different scenarios based on the robot's behavior in order to assure the robustness of the proposed method. A successful trial is considered when the robot can avoid obstacles without collision and reach the goal point within a limited time.

5.1 Test Environment (E1)

The basic scenario (E1_1) in this environment consists of 10 obstacles in addition to the four external walls as shown in "Fig. 7a". The start point and the goal point are the same as the training environment: S(50, 50), G(450, 250). Applying the proposed path planning method, the robot takes the trajectory shown in "Fig. 7a" to reach the goal successfully without colliding with any obstacles.

Some changes are made to the first scenario which is, to lengthen obs2 in a way that closes the whole passage through which the robot has passed previously and after that reapply the test and observe the robot's performance ("Fig. 7b"). In this scenario, we notice that the robot has followed the same old path, but when it reaches the closed way, it turns and takes another path that leads to the goal.

To make it more complex for the robot, the previous way that represents the path executed by the robot is also closed by lengthening obstacle 11, and the experiment is performed again as shown in scenario E1-3. In this case, the robot tries to follow the first and second succeeding paths until it reaches a closed way and eventually, the robot could find a successful path to the goal point.

The final scenario in this environment E1-4 is achieved by changing the goal point to another point G(150, 200) which is in this case more difficult to reach as shown in scenario E1-4. It is noticed that the robot takes a new path after turning around the goal from the closed side until it successfully reaches the new goal. All these scenarios are depicted in online-resource01.

5.2. Test Environments (E2)

To assure the robustness and the power of the trained agent, different scenarios or mazes are tested, the main changes in these tests are the positions and lengths of the obstacles, and consequently, different shapes of the environments are formed. The start and the goal points have been also changed. As we can see in "Fig. 8", with all its subfigures, all the trials were successful and all four scenarios are depicted in online-resource02.

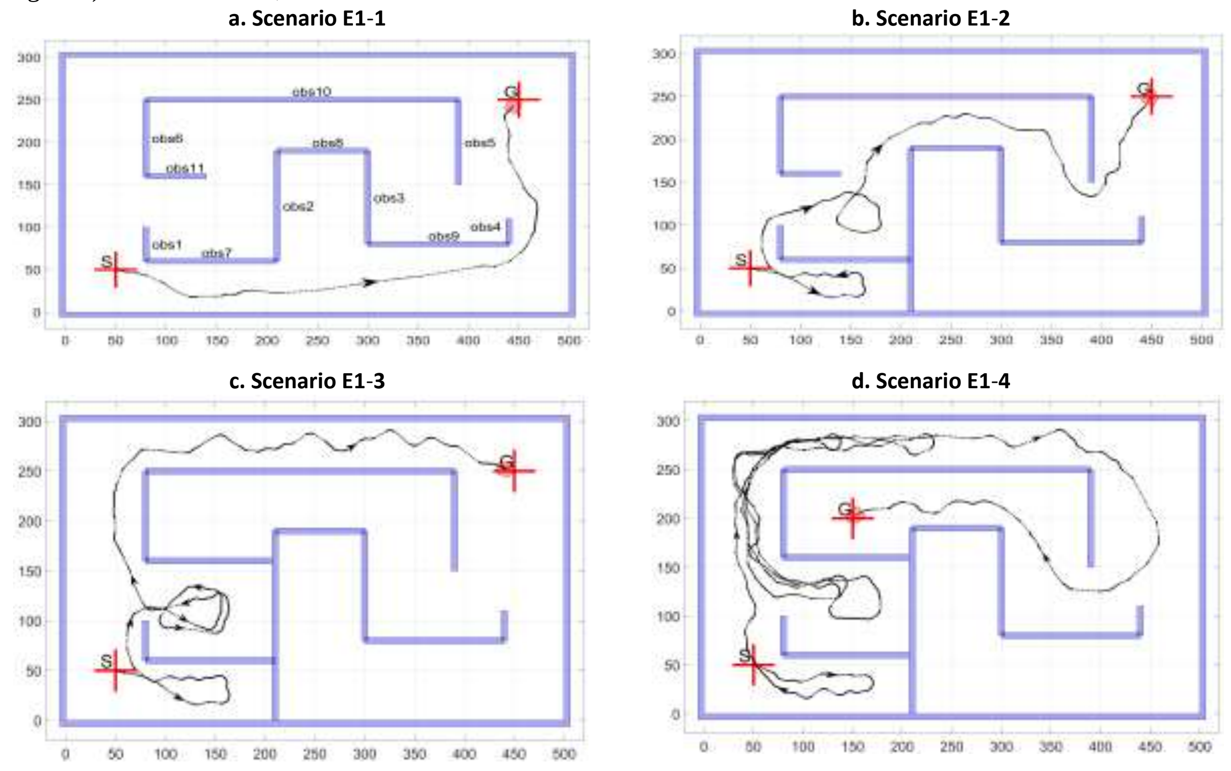

Fig. 7 Solving path planning for different scenarios in the test environment (E1).

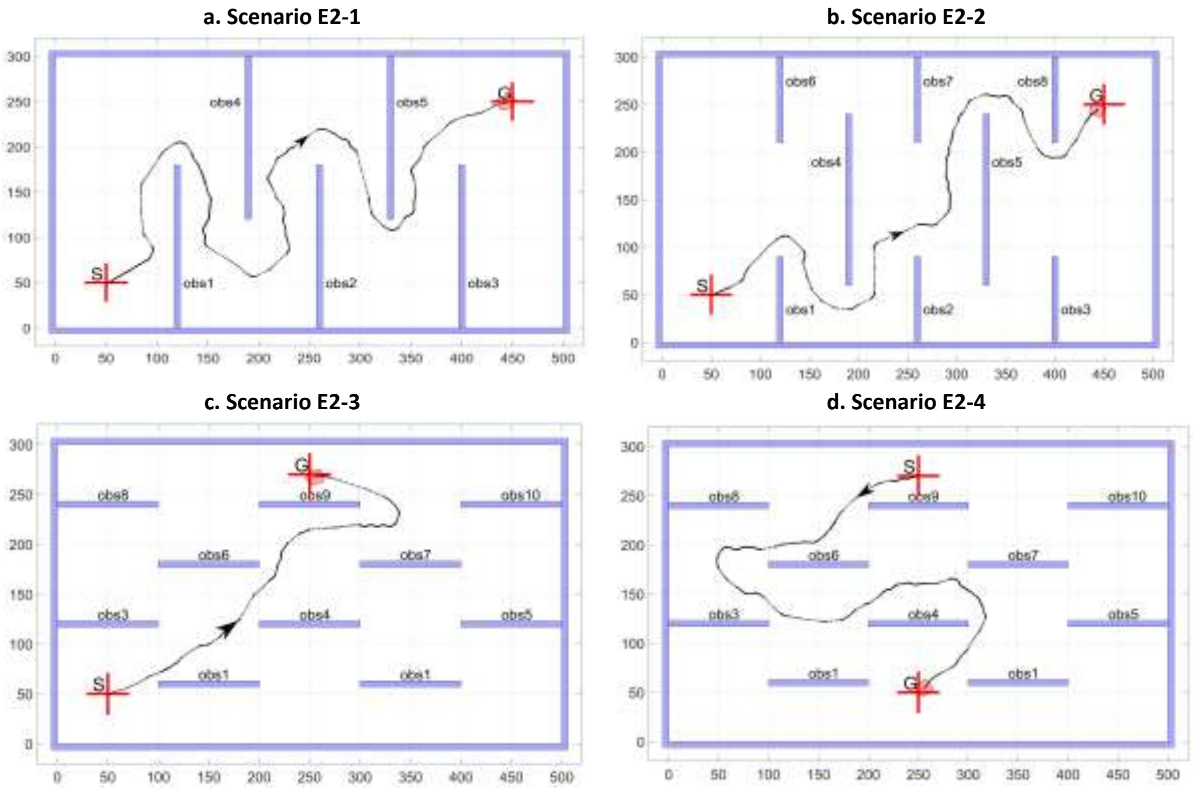

Fig. 8 Solving path planning for different scenarios and different start and goal points in the test environment (E2).

5.3. Test in Real Environment

Real experiments that apply the new approach are carried out using the TurtleBot3 Burger robot shown in "Fig. 9". The TurtleBot3 Burger is a small, programmable, ROS-based (Robot Operating System) mobile robot for use in education, research, and product prototyping. It is a Two-wheeled diff erential drive type platform.

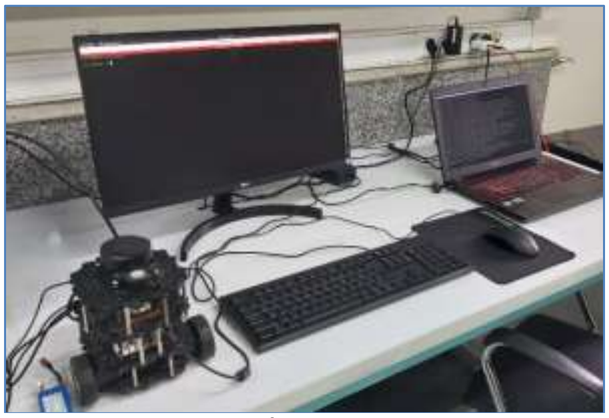

Fig. 9 The robot used for experimental tests (turtlebot3 burger), and the remote laptop used to control the robot.

The TurtleBot3 Burger is equipped with two servo motors, an OpenCR ARM Cortex-M7 control board, and a Raspberry Pi3 embedded computer. In addition to that it includes a gyroscope, accelerometer, 3-axis magnetometer, and 360 deg LiDAR.

The trained neural network with the obtained parameters of the reward has been fed up to TurtleBot3 Burger robot through a ROS2 code. Then, experiments were done to evaluate the proposed approach and to show its ability to be executed on real robots. The test environments shown in "Fig. 10" have a dimension of (250 × 200) cm. The test is done in a completely new environment in which the robot can successfully move from the start point and reach the goal point. The path is shown completely in online-resource3.

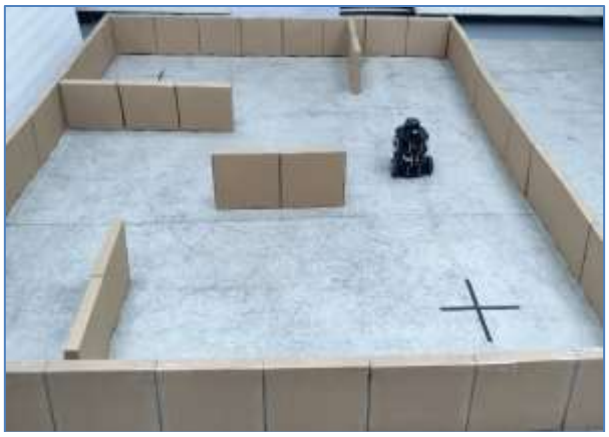

Fig. 10 Path planning test in the real environment.

6 CONCLUSIONS

A new reward function is defined to solve the path-planning problem of a mobile robot working in an unknown dynamic environment. This reward consists of five weighted parts. The coefficients of this reward definition had to be determined and optimized to achieve the most path-planning successful algorithm. Because the training process takes a very long time, a DoE approach is adopted to decrease the number of training operations. This statistical study leads to determining the most effective coefficients of the reward and determining their values. These coefficients are the ones related to the distance and the distance change to the obstacles. Finally, experimental tests of path planning were conducted to prove the efficiency of the trained agent with the obtained parameters and the results show successful training values.

REFERENCES

[1] Zhou, C., Huang, B., and Fränti, P., A Review of Motion Planning Algorithms for Intelligent Robots, Journal of Intelligent Manufacturing, Vol. 33, No. 2, 2022, pp. 387-424, 2022/02/01 2022.

[2] Patle, B., Pandey, A., Parhi, D., and Jagadeesh, A., A Review: on Path Planning Strategies for Navigation of Mobile Robot, Defence Technology, 2019.

[3] Li, Q. L., Song, Y., and Hou, Z. G., Neural Network Based Fastslam for Autonomous Robots in Unknown Environments, Neurocomputing, Vol. 165, 2015, pp. 99-110.

[4] Hart, F., Okhrin, O., Enhanced Method for Reinforcement Learning Based Dynamic Obstacle Avoidance by Assessment of Collision Risk, Neurocomputing, Vol. 568, 2024, pp. 127097.

[5] Lamini, C., Benhlima, S., and Elbekri, A., Genetic Algorithm Based Approach for Autonomous Mobile Robot Path Planning, Procedia Computer Science, Vol. 127, 2018, pp. 180-189.

[6] Abiyev, R., Ibrahim, D., and Erin, B., Navigation of Mobile Robots in The Presence of Obstacles, Advances in Engineering Software, Vol. 41, No. 10-11, 2010, pp. 1179-1186.

[7] Mohanty, P. K., Parhi, D. R., Optimal Path Planning for a Mobile Robot Using Cuckoo Search Algorithm, Journal of Experimental & Theoretical Artificial Intelligence, Vol. 28, No. 1-2, 2016, pp. 35-52.

[8] Cebollada, S., Payá, L., Flores, M., Peidró, A., and Reinoso, O., A State-of-The-Art Review on Mobile Robotics Tasks Using Artificial Intelligence and Visual Data, Expert Systems with Applications, Vol. 167, 2021, pp. 114195.

[9] Bailey, R., Reiss, J., Design and Analysis of Experiments Testing for Biodiversity Effects in Ecology, Journal of Statistical Planning and Inference, Vol. 144, 2014, pp. 69-80.

[10] Montgomery, D. C., Design and Analysis of Experiments, John Wiley & Sons, 2017.

[11] Chen, S., Wang, D., Zuo, A., Chen, Z., Li, W., and Zan, J., Vehicle Ride Comfort Analysis and Optimization Using Design of Experiment, in 2010 Second International Conference on Intelligent Human-Machine Systems and Cybernetics, Vol. 1, 2010, pp. 14-18, IEEE.

[12] Hussain, M., Gu, J., Engel, R., and Shortt, D., Designed Experiment to Find the Optimal Combination of The Factors for The Coordinate Measuring Machine (CMM) to Measure Cylindricity of Engine Cylinder Bore, in 2015 International Conference on Industrial Engineering and Operations Management (IEOM), 2015, pp. 1-8, IEEE.

[13] Zou, G., Xu, J., and Wu, C., Evaluation of Factors that Affect Rutting Resistance of Asphalt Mixes by Orthogonal Experiment Design, International Journal of Pavement Research and Technology, Vol. 10, No. 3, 2017, pp. 282-288.

[14] Hu, Q., Liu, Y., Zhang, T., Geng, S., and Wang, F., Modeling the Corrosion Behavior of Ni-Cr-Mo-V High Strength Steel in the Simulated Deep Sea Environments Using Design of Experiment and Artificial Neural Network, Journal of Materials Science & Technology, Vol. 35, No. 1, 2019, pp. 168-175.

[15] Dufreneix, S., and et al., Design of Experiments in Medical Physics: Application to the AAA Beam Model Validation, Physica Medica, Vol. 41, 2017, pp. 26-32.

[16] Mondragón, J. E., García, J. A. J., Flores, J. M. M., López, A. V., and Vázquez, S. T., Experiments Simulation and Design to Set Traffic Lights' Operation Rules, Transport policy, Vol. 67, 2018, pp. 2-12.

[17] Lv, S., Niu, Z., Cui, Q., He, Z., and Wang, G., Reliability Improvement through Designed Experiments with Random Effects, Computers & Industrial Engineering, Vol. 112, 2017, pp. 231-237.

[18] Qi, H., Osterman, M., and Pecht, M., Design of Experiments for Board-Level Solder Joint Reliability of PBGA Package Under Various Manufacturing and Multiple Environmental Loading Conditions, IEEE Transactions on Electronics Packaging Manufacturing, Vol. 32, No. 1, 2009, pp. 32-40.

[19] Nowzari, R., Mirzaei, N., and Aldabbagh, L., Finding the Best Configuration for a Solar Air Heater by Design and Analysis of Experiment, Energy Conversion and Management, Vol. 100, 2015, pp. 131-137.

[20] Vieira, L. W., et al., Methodology for Ranking Controllable Parameters to Enhance Operation of a Steam Generator with a Combined Artificial Neural Network and Design of Experiments approach, Energy and AI, Vol. 3, 2021, pp. 100040.

[21] Saidi, M., Yousefi, M., Minbashi, M., and Ameri, F. A., Catalytic Upgrading of 4-Methylaniosle as a Representative of Lignin-Derived Pyrolysis Bio-Oil: Process Evaluation and Optimization Via Coupled Application of Design of Experiment and Artificial

Neural Networks, International Journal of Hydrogen Energy, Vol. 46, No. 12, 2021, pp. 8411-8430.

[22] Zhou, Y., Van Kampen, E. J., and Chu, Q., Hybrid Hierarchical Reinforcement Learning for Online Guidance and Navigation with Partial Observability, Neurocomputing, Vol. 331, 2019, pp. 443-457.

[23] Duguleana, M., Mogan, G., Neural Networks Based Reinforcement Learning for Mobile Robots Obstacle Avoidance, Expert Systems with Applications, Vol. 62, 2016, pp. 104-115.

[24] Jaradat, M. A. K., Al-Rousan, M., and Quadan, L., Reinforcement Based Mobile Robot Navigation in Dynamic Environment, Robotics and Computer-Integrated Manufacturing, Vol. 27, No. 1, 2011, pp. 135-149.

[25] Xia, C., El Kamel, A., Neural Inverse Reinforcement Learning in Autonomous Navigation, Robotics and Autonomous Systems, Vol. 84, 2016, pp. 1-14.

[26] Smart, W. D., Kaelbling, L. P., Effective Reinforcement Learning for Mobile Robots, in Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292), Vol. 4, 2002, pp. 3404-3410, IEEE.

[27] Wei, M., Wang, S., Zheng, J., and Chen, D., UGV Navigation Optimization Aided by Reinforcement Learning-Based Path Tracking, IEEE Access, Vol. 6, 2018, pp. 57814-57825.