Dynamic Obstacle Avoidance for Mobile Robots Using Vision-Based Deep Reinforcement Learning

Kagan Yavuz
Istanbul Technical University, Faculty of Electrical and Electronics, Control Engineering
Istanbul, Turkey

***Abstract*—This study addresses dynamic obstacle avoidance for mobile robots using vision-based deep reinforcement learning (DRL). A Double Deep Q-Network (D3QN) is employed to enable a TurtleBot3 robot to navigate in environments containing both static and dynamic obstacles, using synchronized RGB and depth images as sensory inputs. Unlike prior work that relies on LIDAR and depth estimation through GANs, this research directly utilizes depth cameras, simplifying the setup and improving perception accuracy. The Gazebo simulator and Robot Operating System (ROS) were used for implementation and testing. The model was trained on datasets capturing diverse scenarios, including 10,000 images for static, dynamic, and combined obstacles. Experimental results demonstrate that models trained on dynamic obstacle datasets performed better in generalization and adaptability. Future extensions include integrating goal-directed navigation and additional sensory modalities for improved real-world applicability.**

Keywords— Deep Reinforcement Learning; Obstacle Avoidance; TurtleBot3; Robot Operating System (ROS); Gazebo; Double Deep Q-Network (D3QN); Mobile Robots; OpenAI

I. INTRODUCTION

Robotic navigation in dynamic environments is a critical challenge in the field of autonomous systems. Mobile robots must navigate complex surroundings while avoiding static and moving obstacles, ensuring safety and efficiency. Vision-based navigation, leveraging RGB and depth images, offers significant potential in addressing these challenges by providing rich sensory input for decision-making. The advent of deep reinforcement learning (DRL) has further revolutionized robotic control, enabling agents to learn optimal policies through interaction with their environment.

This study presents a vision-based dynamic obstacle avoidance framework for a TurtleBot3 robot in a simulated environment. The primary objective is to train the robot to navigate efficiently while avoiding collisions with both static and moving obstacles. Unlike prior studies that often focus solely on static environments, this work incorporates dynamic obstacles, thereby enhancing the adaptability of the robot to real-world scenarios.

The proposed approach utilizes a Double Deep Q-Network (D3QN), an advanced DRL algorithm. This architecture is particularly suited for robotic control as it separates state-value and action-advantage estimations, mitigating overestimation issues common in reinforcement learning. A monocular RGB camera and a depth camera mounted on the robot provide synchronized visual data for training and testing. The dataset comprises images collected in static and dynamic environments, offering diverse scenarios for robust policy learning.

This work is implemented using ROS and Gazebo, with TensorFlow for the D3QN model. By training on datasets prepared specifically for static obstacles, dynamic obstacles, and a combination of both, and testing these different models in Gazebo, we demonstrate the effectiveness of the model in dynamic obstacle-rich environments. This study lays a foundation for extending DRL-based obstacle avoidance to real-world mobile robotics applications. The image captured by TurtleBot3's RGB camera can be seen in Figure 1.

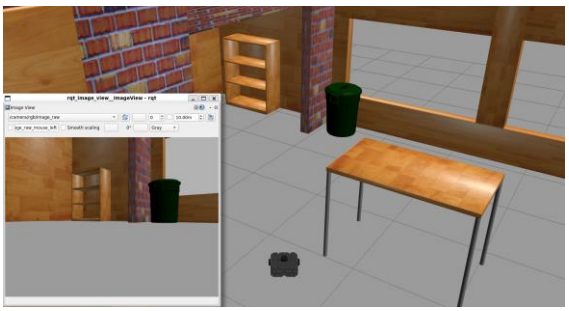

Figure 1 TurtleBot3 with RGB camera in the turtlebot3_house world

II. DEFINITION AND IMPORTANCE OF THE PROBLEM

Dynamic obstacle avoidance is a crucial capability for mobile robots operating in real-world environments. Unlike static navigation scenarios, dynamic environments require robots to respond to moving objects and changing conditions in real-time. Vision-based solutions have gained prominence due to their ability to capture detailed spatial and contextual information through RGB and depth images. However, leveraging this information for effective navigation necessitates advanced decision-making frameworks, such as those offered by deep reinforcement learning.

In our approach, the navigation problem is formulated as a reinforcement learning task where the robot must learn to map visual observations to optimal actions. The problem can be formally defined as follows:

1. State (s_t) : At each timestep t, the state consists of an RGB image (I_{RGB}) and a depth image (I_{Depth}), concatenated into a tensor:

$$s_t = concat(I_{Depth}, I_{RGB})$$

Here, $I_{RGB} \in R^{HxWx3}$ and $I_{Depth} \in R^{HxWx1}$, where H and W are the height and width of the images.

2. Action (a_t): The robot has a discrete set of actions:

$$a_t \in \{a_0, a_1, \dots, a_6\}$$

Each action corresponds to a specific motion command, such as forward movement, left or right turns, or stopping.

3. Reward (r_t): A dense reward function incentivizes forward movement and penalizes collisions or unsafe navigation. The reward at time t is defined as:

$$r_t = \begin{cases} -10 & if\ collusion\ occurs \\ +1 & for\ forward\ motion \\ 0 & otherwise \end{cases}$$

4. Objective: The agent aims to maximize the cumulative discounted reward (R) over an episode:

$$R = \sum_{t=0}^{T} \gamma^t r_t$$

where T is the episode length and γ is the discount factor.

The Double Deep Q-Network (D3QN) is employed to approximate the action-value function $Q(s, a)$, enabling the agent to select the optimal action for each state.

The deep reinforcement learning agent is trained within the ROS-Gazebo simulation environment, incorporating the ros-gazebo-gym toolkit. The agent learns to predict optimal actions based on visual input, enabling it to avoid obstacles dynamically.

The primary objectives of this project are:

1. To design and implement a deep reinforcement learning framework for dynamic obstacle avoidance.
2. To evaluate the framework's performance in navigating TurtleBot3 within dynamic environments.
3. To compare the proposed approach against baseline methods and static obstacle setups to validate its efficacy.

By addressing the challenges of dynamic obstacle avoidance, this project aims to contribute to the development of robust and adaptive navigation systems for mobile robots in real-world scenarios

III. Related Studies

The ability of mobile robots to autonomously navigate through complex and dynamic environments has broad implications for various real-world applications, including autonomous vehicles, warehouse automation, search and rescue missions, and healthcare delivery systems. Dynamic obstacle avoidance is a critical capability for ensuring the safety and efficiency of these systems, particularly in scenarios where environments are unpredictable and continuously changing. Deep reinforcement learning (DRL), which enables agents to learn optimal policies from their interactions with the environment, has emerged as a powerful tool for addressing such challenges. By integrating vision-based sensing and DRL, autonomous robots can perceive and adapt to their surroundings in a more robust and scalable manner.

A prominent study that serves as the foundation for this project is the work by Patrick Wenzel, Torsten Schön, Laura Leal-Taixé, and Daniel Cremers, titled "Vision-Based Mobile Robotics Obstacle Avoidance With Deep Reinforcement Learning" (2020). The authors present a DRL-based approach for enabling a TurtleBot2 robot to avoid obstacles in simulated environments using visual inputs. Their method employs a monocular RGB camera as the primary sensory input, capturing observations of the environment to feed into a deep reinforcement learning model. The reinforcement learning framework is implemented using the gym-gazebo toolkit, which integrates Gazebo simulation environments with OpenAI's Gym interface to facilitate training and testing.

The study focuses on training the robot in two specific environments: circuit2-v0, a narrow, corridor-like path, and maze-v0, a complex, maze-like structure. The reward function is designed to incentivize collision-free navigation and penalize actions that lead to failures, such as collisions or deviations from the intended path. The trained model demonstrates effective obstacle avoidance in the predefined static environments.

While the study achieves notable success in addressing static obstacle scenarios, it exhibits limitations in adapting to dynamic environments, where obstacles exhibit motion and unpredictability. The reliance on static training environments (circuit2-v0 and maze-v0) restricts its applicability in real-world situations.

Building upon this study, the proposed project aims to address these limitations by introducing dynamic moving obstacles into the training and testing environments and using TurtleBot3 for enhanced capabilities. This extension is expected to improve the robustness and applicability of vision-based obstacle avoidance in real-world scenarios.

IV. Research Procedure

The proposed study employs deep reinforcement learning (DRL) as the core methodology to address the problem of dynamic obstacle avoidance in mobile robotics. DRL enables the development of an end-to-end learning framework where raw sensory data is directly mapped to control commands through the optimization of a policy. A monocular RGB camera and a depth camera will serve as the primary sensory inputs, capturing visual data to perceive the robot's environment. The visual data will be processed to extract features and inform the decision-making process, ensuring efficient and collision-free navigation in dynamic settings.

The simulation and experimentation will be conducted using the Robot Operating System (ROS) integrated with Gazebo, a widely used robot simulation environment. These tools provide

a flexible platform for developing, testing, and evaluating robot behavior under controlled conditions. The ros-gazebo-gym toolkit will be employed to facilitate the implementation of reinforcement learning algorithms. This toolkit bridges the gap between ROS and DRL frameworks, enabling seamless integration of the learning process into the simulation environment.

The research methodology builds upon a vision-based obstacle avoidance framework, significantly diverging from the approach outlined in the reference study, "Vision-Based Mobile Robotics Obstacle Avoidance With Deep Reinforcement Learning" by Patrick Wenzel et al. (2020). The key enhancements include the use of a monocular RGB camera paired with a depth sensor, the incorporation of dynamic obstacles, and the adoption of an advanced Double Deep Q-Network (D3QN) model. Below is a detailed explanation of the research procedure:

A Simulation Environment and Tools

The experiments are conducted in a simulated environment using ROS Noetic and Gazebo 11. The TurtleBot3 platform is selected for its compatibility with the TurtleBot3_house world, a map consisting of both static obstacles (walls, furniture) and dynamic elements (e.g., a moving ball introduced in this study). These tools provide a high-fidelity simulation environment that closely replicates real-world robotic navigation. TurtleBot3 and the turtlebot3_house world are shown in Figure 2.

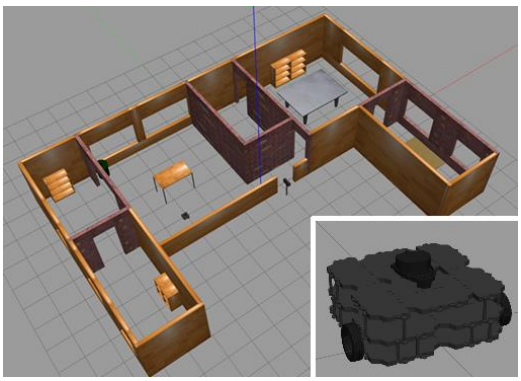

Figure 2 TurtleBot3 and the turtlebot3_house world

B Dataset Collection

To train the D3QN model effectively, a dataset of synchronized RGB and depth images was created, utilizing direct depth measurements from the robot's onboard depth camera, in contrast to the reference study that estimates depth data using GANs. The dataset captures real-world navigation complexities by including dynamic obstacles, which introduce variability in motion and position. Two separate datasets were created and trained independently: one consisting of 5,000 RGB and 5,000 depth images with only static obstacles, and the other containing 5,000 RGB and 5,000 depth images with both static and dynamic obstacles. After evaluating the results, the datasets were merged into a combined dataset of 10,000 RGB and 10,000 depth images featuring both static and dynamic obstacles, and a new model was trained and tested. The outcomes of these three separate experiments will be discussed in the following sections. Various RGB and depth images from the created dataset are presented in Figure 3.

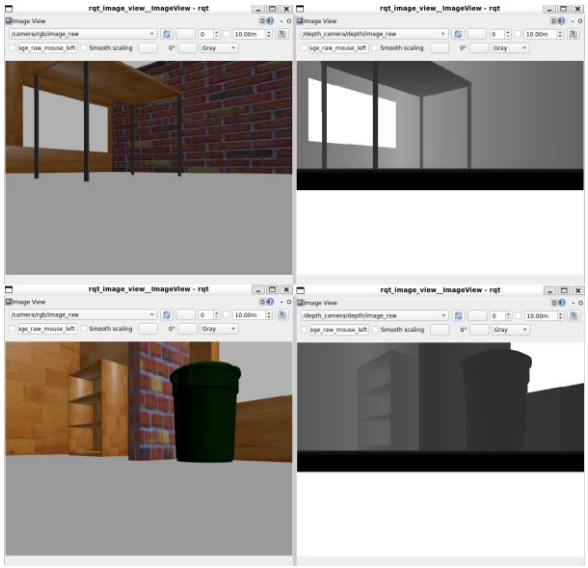

Figure 3 Table RGB and depth images (Top), shelf and trash can RGB and depth images (Bottom)

Images are collected at a resolution of 128×160, with the message_filters library ensuring temporal alignment between RGB and depth data. The synchronized images are stored in separate directories for dynamic and static scenarios, forming a diverse dataset for training and testing.

C Deep Reinforcement Learning Model

The navigation problem is formulated as a Deep Q-Learning task. Unlike the reference study that employs a standard Deep Q-Network (DQN), this research utilizes D3QN to address overestimation issues in Q-value predictions. D3QN improves upon DQN by decomposing the Q-value into:

$$Q(s,a) = V(s) + \left(A(s,a) - \frac{1}{|A|} \sum_{a'} A(s,a') \right)$$

Here, $V(s)$ represents the state value, while $A(s,a)$ captures the advantage of taking action a in state s. This decomposition ensures more stable and accurate policy learning compared to DQN.

D Training and Testing Procedure

The model is trained over 10 episodes for each dataset containing static and dynamic images and 20 episodes for mixed dataset, with each episode consisting of steps equal to the dataset size. Training incorporates dense rewards to incentivize forward motion and penalize collisions. The trained model is saved periodically, generating .keras files representing the learned policies.

For testing, the model is deployed in Gazebo with the TurtleBot3_house world. The trained policy is evaluated by navigating the robot through dynamic and static obstacle scenarios, ensuring collision-free motion. Actions are inferred

using the D3QN model, and their effectiveness is validated based on navigation success rates.

E Comparison with Reference Study

While the reference study employs DQN and combines LIDAR with monocular RGB data, this project relies on synchronized RGB and depth images from the rgb and depth cameras attached to TurtleBot3. This simplifies the setup and maintains robust obstacle avoidance.

The reference study uses GANs to predict depth images, adding computational overhead and potential inaccuracies. Here, depth is directly obtained from a depth camera, providing accurate and real-time depth perception.

Simulation environments also differ. The reference study uses static obstacle environments, while this project employs the TurtleBot3_house map with dynamic obstacles in Gazebo, offering training in more realistic scenarios.

Lastly, the architecture is advanced from DQN to D3QN, reducing overestimations and enhancing stability, leading to better performance in complex environments.

V. EXPERIMENTAL RESULTS

This section evaluates the performance of the trained models in various scenarios, using datasets containing RGB and depth images with static, dynamic, and combined obstacles. Each experiment is designed to analyze the robot's ability to navigate safely and effectively while avoiding collisions.

A Static Obstacles: models_static

The dataset consists of 5000 RGB and 5000 depth images featuring static obstacles such as walls and furniture in the TurtleBot3_house map. Episode value was set to 5 and a new model saved at the end of every episode. Total rewards and loss values for all models created can be seen in Table 1.

Table 1 model_static Training Results

Episode Number	Total Reward	Loss
1/5	2468.96	0.0281
2/5	2487.72	0.0384
3/5	2498.57	0.0360
4/5	2495.56	0.0451
5/5	2490.84	0.0307

Among these, the model from episode 3 demonstrated the highest performance with a total reward of 2,498.57 and a loss of 0.0360. This model successfully navigated through the environment without collisions, highlighting its potential for reliable obstacle avoidance.

However, the model from episode 5, which achieved a total reward of 2,490.84 and a lower loss of 0.0307, encountered challenges such as colliding with table legs. After a collision, the robot could not recover, as the obstacle moved out of the camera's field of view, causing it to perceive the path ahead as clear. This scenario underscored the limitations of the vision-only approach in detecting small or partially visible obstacles. This situation can be seen in Figure 4.

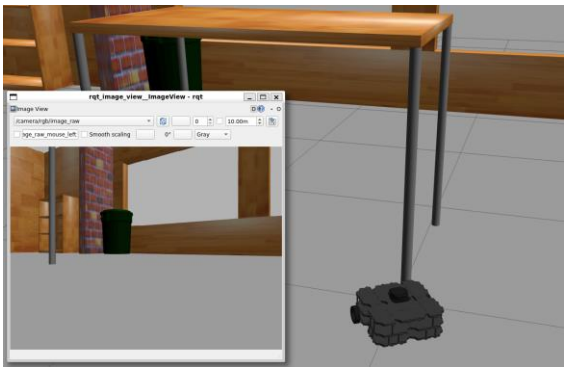

Figure 4 Although TurtleBot3 collided with the table leg, it took an incorrect action, as the camera perceived the path ahead to be clear.

Memory usage was a significant challenge in this experiment, with RAM utilization reaching 94%, causing a slowdown in training. To address this issue, the MEMORY_SIZE was reduced from 10,000 to 5,000, and the BATCH_SIZE was decreased from 32 to 16, improving the model's efficiency while preserving training quality.

B Dynamic Obstacles: models_dynamic

In the dynamic obstacle scenario, the dataset contained 5000 RGB and 5000 depth images with moving obstacles, such as a ball introduced into the environment. The training parameters remained consistent with those of the static obstacle experiment. 10 models were trained, the training results can be seen in Table 2.

Table 2 model_dynamic Training Results

Episode Number	Total Reward	Loss
1	2522.90	0.0365
2	2471.31	0.0843
3	2518.57	0.0531
4	2496.28	0.0242
5	2514.92	0.0217
6	2486.74	0.0343
7	2497.64	0.0333
8	2478.89	0.0580
9	2487.41	0.0290
10	2482.43	0.0552

It can be seen that the model from episode 5 achieving the best results. This model recorded a total reward of 2,514.92 and a loss of 0.0217, demonstrating superior performance compared to models trained on static obstacles. The robot navigated static obstacles effectively, colliding with only one wall due to a misinterpretation of the camera feed when the wall moved out

of view. Remarkably, the model excelled at avoiding dynamic obstacles, successfully stopping and waiting for the moving ball to pass before maneuvering around it. These results indicate that the inclusion of dynamic obstacles in the dataset improved the model's generalization capabilities. Training on dynamic scenarios equipped the robot to handle unpredictable environments, a critical feature for real-world applications. In Figure 5, the moving ball and its corresponding RGB and depth images can be observed.

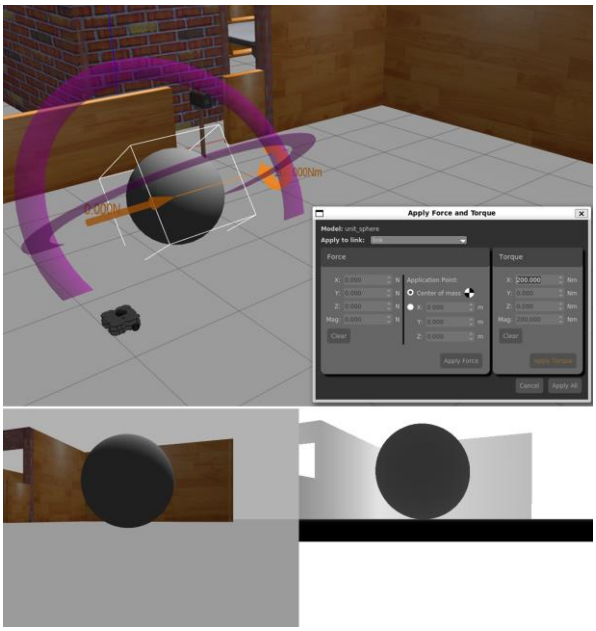

Figure 5 Moving obstacles are added to the world while training

C Dynamic & Static Obstacles: models_dynamic_and_static

In the combined obstacle scenario, the dataset was a mixture of static and dynamic obstacle images, with 5000 RGB and 5000 depth images for each type. Duration of training chosen as 20 episodes, with a model saved every two episodes. The training results can bee seen in Table 3.

Table 3 model_dynamic_and_static Training Results

Episode Number	Total Reward	Loss
1-2	2498.82	0.0703
3-4	2503.17	0.0492
5-6	2498.47	0.0278
7-8	2489.67	0.0209
9-10	2511.87	0.0302
11-12	2503.44	0.0316
13-14	2498.03	0.0248
15-16	2504.68	0.0238
17-18	2472.46	0.0291
19-20	2521.41	0.0182

The best-performing model was derived from episodes 19 and 20, with an average total reward of 2,521.41 and a loss of 0.0182. This model successfully navigated around static obstacles like tables but occasionally collided with walls. Once again, recovery from collisions was problematic when the robot perceived empty space after close-range impacts. While the model demonstrated satisfactory performance with static obstacles, its effectiveness against dynamic obstacles was less impressive compared to models trained exclusively on dynamic scenarios.

Another notable model from episodes 9 and 10 achieved an average total reward of 2,511.87 and a loss of 0.0302, but it exhibited a higher rate of collisions with both static and dynamic obstacles. These results suggest that combining datasets may dilute the model's ability to specialize in either static or dynamic scenarios, indicating the need for a balanced dataset or scenario-specific training to optimize performance.

VI. Possible Optimizations and Conclusion

Several general observations emerged from the experiments. First, the robot struggled with obstacles that were partially or temporarily outside the camera's field of view, such as table legs or walls after collisions. These issues highlight the limitations of a vision-only approach in detecting and recovering from such scenarios. Second, the absence of a goal-oriented navigation framework caused the robot to exhibit behavior that lacked purpose, focusing solely on avoiding obstacles without aiming for a specific destination. Integrating a target location into the navigation task could enhance the robot's functionality and realism. Third, the inability to recover from collisions after perceiving empty space stems from the robot's reliance on camera data alone. Close-range impacts resulted in the misclassification of the environment as obstacle-free. Adding a LIDAR sensor during training could address this limitation, enabling more accurate detection of surfaces at close distances. Finally, the use of larger and more diverse datasets, combined with advanced training techniques, could further improve the model's generalization capabilities and robustness in real-world scenarios.

These experiments underscore the effectiveness of the D3QN architecture in vision-based navigation while identifying critical areas for improvement. The findings provide valuable insights into the design and implementation of robust robotic navigation systems in complex environments.

VII. References

Zamora, I., Lopez, N. G., Vilches, V. M., & Cordero, A. H. (2017). Extending the OpenAI gym for robotics: A toolkit for reinforcement learning using ROS and Gazebo. *arXiv preprint arXiv:1608.05742v2*.

Rickstaa. (n.d.). RICKSTAA/Ros-gazebo-gym: Framework for integrating ROS and Gazebo with Gymnasium, streamlining the development and training of RL algorithms in realistic robot simulations. *GitHub*. Retrieved January 27, 2025, from https://github.com/rickstaa/ros-gazebo-gym

Robotis-Git. (n.d.). Robotis-git/TurtleBot3: ROS packages for TurtleBot3. *GitHub*. Retrieved January 27, 2025, from https://github.com/ROBOTIS-GIT/turtlebot3

Wenzel, P., Schon, T., Leal-Taixe, L., & Cremers, D. (2021). Vision-based mobile robotics obstacle avoidance with deep reinforcement learning. In *2021 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 14360–14366). https://doi.org/10.1109/icra48506.2021.9560787

Xie, L., Wang, S., Markham, A., & Trigoni, N. (2017). Towards monocular vision-based obstacle avoidance through deep reinforcement learning. In *RSS 2017 Workshop on New Frontiers for Deep Learning in Robotics*. Retrieved January 27, 2025, from http://dblp.uni-trier.de/db/journals/corr/corr1706.html#XieWMT17

Xie, L. (n.d.). Xie9187/Monocular-Obstacle-Avoidance: Codebase for "Towards Monocular Vision Based Obstacle Avoidance through Deep Reinforcement Learning." *GitHub*. Retrieved January 27, 2025, from https://github.com/xie9187/Monocular-Obstacle-Avoidance