Sona Preethy Wilson

Analysis and Improvement of Deep Reinforcement Learning-Based Mobile Robot Navigation in Dynamic Environments

School of Technology and Innovations
Master's in Smart Energy
Robotics

Vaasa 2026

UNIVERSITY OF VAASA
School of Technology
Author: Sona Preethy Wilson
Title of the Thesis: Analysis and Improvement of Deep Reinforcement Learning-Based Mobile Robot Navigation in Dynamic Environments
Degree: Master of Science in Technology
Programme: Master's in Smart Energy
Major: Robotics
Supervisor: Elham Ahmadi
Thesis Evaluator: Jani Boutellier
Year: 2026 **Pages:** 79

ABSTRACT:

Autonomous mobile robot navigation in dynamic environments has been studied widely because of the increasing necessity for safe and efficient movement in complex settings such as warehouses, hospitals, transport terminals, and public spaces. Even though classical navigation approaches have shown stronger performance in controlled static environments, real-world continuous applications require repeated replanning, map updates, and simplified assumptions about obstacle position and dynamics. These requirements often limit the application of classical approaches in complex scenarios. Therefore, deep reinforcement learning is progressively adopted as a promising approach for navigation tasks requiring continuous control and adaptation to changing environments.

In this thesis, deep reinforcement learning-based mobile robot navigation in dynamic environments has been analyzed and improved with a focus on safety, stability, task completion, and generalization. The research problem is aligned with the difficulty of learning reliable navigation policies, mainly in dynamic environments, where the performance can be reduced due to sparse rewards, unsafe exploration, and limited transferability to unseen settings. The study has been rooted in reinforcement learning theory, the Markov decision processes, continuous control, and safety-aware reward design function. Notably, DDPG, TD3, and SAC have been used as proposed algorithms for continuous control.

Using structured navigation environments, a simulation-based experimental framework has been developed. The baseline DDPG has first been evaluated and conducted both qualitative and quantitative analysis on its failure behavior. After this, an iterative reward refinement was performed to improve collision avoidance, goal reaching, and motion stability. Later, algorithmic enhancement was performed, where TD3 and SAC were compared under controlled static and dynamic settings, and SAC was selected for curriculum-based training because of its generalization behavior. Finally, a curriculum-based training has been implemented with progressive complex environments followed by three generalization assessments, including zero-shot evaluation, warm-up fine-tuning, and OOD stress sweep.

The results show that the improved model is more stable and has effective navigation behavior. The curriculum-trained policy achieved consistent task completions and has shown reasonable transferability to unseen environments, although performance degrades in more difficult out-of-distribution cases. Overall, the study has demonstrated that a combination of reward engineering, stable learning, and curriculum-based training can improve safe navigation in dynamic environments.

KEYWORDS: deep reinforcement learning, mobile robot navigation, dynamic environments, curriculum learning, SAC, TD3, generalization

Contents

Figures

Tables

Abbreviations

CR: Collision Rate

DRL: Deep Reinforcement Learning

DDPG: Deep Deterministic Policy Gradient

LiDAR: Light Detection and Ranging

MDP: Markov Decision Process

OOD: Out-of-distribution

PPO: Proximal Policy Optimization

RL: Reinforcement Learning

SAC: Soft Actor-Critic

SR: Success rate

TD3: Twin-Delayed Deep Deterministic Policy Gradient

WU: Warm-Up

ZS: Zero-Shot

1 Introduction

1.1 Background

In a real-world dynamic environment like warehouses, hospitals, public stores, transportation terminals, and urban sidewalks, the implementation of autonomous mobile robots is increasing currently. This demands the need for optimized decision-making while maintaining the balance between safety and efficiency in a complex environment (Rudenko et al., 2020). Also, collisions are unacceptable considering the comfort and safety of the pedestrians. Hence, there exists a challenge in mobile robotics for designing a socially acceptable, safe, efficient, and robust navigation system.

Traditional navigation methods include Simultaneous Localization and Mapping (SLAM) and classical planners such as A*, Dijkstra, Artificial Potential Field (APF), Rapidly-exploring Random Tree (RRT), and Dynamic Window Approach (DWA) for mapping, localization, path planning, and obstacle avoidance. These methods perform well in static, well-known environments but fail to adapt to dynamic scenarios (Zeng et al., 2021). This happens because navigation in dynamic scenarios depends on recurrently updated maps, constant replanning, and handling of dynamic obstacles. In most of the systems, dynamic objects like pedestrians and other moving objects are considered static, or an oversimplified assumption of their movement often leads to dangerous decisions in a complex environment.

These constraints encourage exploring the learning-based approaches as they can connect the sensor inputs, varying environments, and safety measures. Deep Reinforcement Learning (DRL) combines deep learning with reinforcement learning, where the robot learns by interacting with the environment (Lillicrap et al., 2016). With the integration of deep neural networks, DRL has been deployed in navigation systems for obstacle avoidance, indoor, and social navigation, as DRL can estimate policies and value functions in high-dimensional state spaces.

For discrete actions, value-based algorithms like DQN, Double DQN, and Dueling DQN, and other variants deployed the first end-to-end navigation policies. Similarly, for continuous control, commonly used for real robots, actor-critic methods like Deep Deterministic Policy Gradient (DDPG), Asynchronous Advantage Actor-Critic (A3C), Proximal Policy Optimization (PPO), Soft Actor-Critic (SAC), and Twin Delayed DDPG (TD3) are used for yielding smooth linear and angular velocities (Schulman et al., 2017; Haarnoja et al., 2018; Fujimoto et al., 2018). Regardless of optimistic results, DRL-based navigation still lacks generalization, sim-to-real gaps, partial observability, and sparse rewards when transferring from simulations to complex real-world environments. These limitations are critical for mobile robot navigation in crowded and dynamic environments.

In addition, most of the surveys reviewed found that many of the DRL methods are simulated in idealized conditions about the sensor noise, obstacle movement, and positions, and evaluated in simplified environments. Simultaneously, they emphasize the open challenges of DRL methods, such as safe behavior in highly crowded environments, sim-to-real transfer robustness, multi-sensor fusion, and managing multi-robot coordination (Zhu et al., 2025; Le et al., 2024). Therefore, there is a strong agreement that analyzing and improving the safety and generalization of DRL-based mobile robot navigation in dynamic environments is a key research area.

Safe Reinforcement Learning (SRL) emerged explicitly addressing safety, where safety objectives are separated from performance objectives (Zhou et al., 2023). Conflict-Averse Safe Reinforcement Learning (CASRL) was introduced recently, where it decomposes tasks into sub-tasks, goal reaching, and collision avoidance. This considerably improves safety in dynamic crowded environments and can be implemented in real robots as it includes reliable safety critics and conflict-averse gradient manipulation.

This thesis focuses on analyzing deep reinforcement learning methods for safe and efficient mobile robot navigation in dynamic environments. Also, analyzes how the integration of safety-critical measures into continuous-control algorithms, such as DDPG, SAC, and TD3, allows safe and efficient mobile robot navigation, contributing to the development of DRL-based navigation systems.

1.2 Problem Statement

The navigation methods have been efficient in controlled settings and known environments. But they face many challenges when exposed to an unknown environment, which reduces their efficiency in real-world robotics deployments. Compared to static environments, learning in dynamic environments with time-dependent obstacles will lead to:

- Higher collision rate
- Lower success rate
- Higher training instability
- Policy that circles the obstacles rather than progressing
- Poor generalization to unseen environments

These limitations occur due to:

- Sparse rewards: Sparse and delayed rewards lead to slow convergence.
- Instability: Variability in the reward signal is introduced by dynamic obstacles, causing an instability in learning.
- Reactive reward design: Agents will learn and make unsafe decisions as the collision penalties are applied only after collisions.
- Limited generalization: Policies often fail to adapt to new layouts or different dynamics other than their trained environment.

Formally, the navigation problem can be developed as a Markov Decision Process (MDP) (Sutton & Barto, 2018) defined by the tuple:

$$M = (S, A, P, R, \gamma)$$

where:

- S represents the state space (robot position, velocity, relative obstacle distance, etc.),
- A represents the continuous action space (e.g., angular and linear velocities),
- P represents the state-transition dynamics,
- R is the reward function with $r_t = R(s_t, a_t, s_{t+1})$,
- $\gamma \, \epsilon(0,1)$ is the discount factor.

The primary objective is to learn an optimal policy $\pi(a|s)$ that maximizes the expected cumulative reward:

$$J(\pi) = E\left[\sum_{t=0}^{T} \gamma^t r_t\right]$$

Problem definition: Given this MDP structure, the main goal is to design a reward-aware and safe navigation learning framework that the learned policy will achieve higher success rates, lower collision rates, and robust behavior in dynamic environments.

Research Problem: The standard DDPG struggles to learn a policy that is reliable in dynamic environments due to insufficient reward design and a deficiency of dedicated safety awareness and obstacle avoidance mechanisms. Therefore, the learned policies often fail to balance efficiency and safety in the presence of non-static obstacles.

1.3 Research Questions

This thesis is guided by the following research questions:

1. What behavioral patterns can be observed in the failed navigation policy during training and evaluation?
2. How does safety-aware reward shaping improve high-collision navigation, instability, and goal-reaching performance?
3. What is the effectiveness of the curriculum-based training across increasing environmental complexity?
4. How does the navigation performance of the trained policy change and generalize across different unseen environments without retraining?

1.4 Research objectives

This thesis aims to analyze and improve deep reinforcement learning-based mobile robot navigation in dynamic environments by implementing iterative safe reward designs, behavior failure analysis, and evaluating robustness across more complex, unseen scenarios.

The objectives of this thesis are:

1. To analyze the behavioral failure patterns of standard DDPG in a simple environment
2. To design and implement a safety-aware reward function to improve stability, wall collision avoidance, and goal-reaching behavior.
3. To implement a curriculum-based training with increasing environmental complexity.
4. To conduct a generalization assessment of the trained policy in unseen environments without retraining.

1.5 Contributions

The key contributions of this thesis are as follows:

- **Behavioral failure analysis:** In contrast to existing literature that focuses mainly on quantitative metrics like success and collision rates, this work conducts a qualitative behavioral failure analysis by tracing agent trajectories and recording failure interactions. This helps to identify the unsafe, inefficient policy patterns that lead to unstable, unreliable navigation.
- **Safety-aware reward design:** This work focuses on a systematic iteration of safety-aware reward shaping. Multiple reward variants are designed and evaluated for their influence on success rate, collision rate, collision avoidance, and task completion. This helps to address the failure behaviors, such as higher collisions, freezing behavior, and unsafe and goal-irrelevant navigation.
- **Systematic curriculum-based validation:** Apart from single-environment training, this thesis focused on a systematic, curriculum-based evaluation framework across six controlled complexity stages. This systematic progression helps assess policy performance and robustness as spatial complexity and dynamic obstacle density increase. Moreover, a generalization assessment by testing the agent in separate, unseen environments without retraining evaluates both zero-shot and warm-up generalization, providing a reliable assessment of the policy's adaptability in real-world scenarios.

2 Literature Review

This chapter systematically reviews the evolution of the autonomous mobile robot navigation from classical navigation approaches, deep reinforcement learning (DRL), and safety-aware navigation methods. The review clearly demonstrates the gaps that motivate this thesis: (1) a lack of behavioural failure analysis, (2) the importance of reward functions in improving the performance, and (3) the absence of evaluation across different environment levels.

2.1 Classical Navigation Method

Autonomous mobile robot navigation in dynamic environments is a significant research area in robotics and artificial intelligence. The traditional navigation approach includes simultaneous localization and mapping (SLAM), path planning, and local obstacle avoidance with the classical planners such as A*, Dijkstra, Rapidly-exploring Random Tree (RRT), Artificial Potential Field (APF), and Dynamic Window Approach (DWA). These systems perform well in static, well-known environments but struggle in dynamic environments, which require repeated planning, map updates, grid-resolution trade-offs, and accumulated errors in the perception-navigation stack (Zeng et al., 2021; Le et al., 2024).

The global path planning algorithms, like A* or Dijkstra, compute the shortest path towards the goal, while local planners deal with obstacle handling and smoother navigation in static environments. These algorithms work on discrete maps and are optimal under appropriate assumptions. The researchers later integrated real-time dynamic methods as the environment complexity increased, and these classical methods struggled with performance and robustness across dynamic environments with moving obstacles or multiple agents (Everett et al., 2021).

Similarly, some models emerged later that adapt within highly crowded environments, such as behaviour-based and social force models. But they require higher parameter tuning and were limited to adapt in higher-level social awareness or among multiple interactive agents (Godoy et al., 2016). Another planner that is more applicable for continuous navigation includes the Rapidly exploring Random Tree (RRT) and RRT*. These methods work in complex

spaces but still struggle in a densely populated environment with obstacles having unpredictable trajectories (Xiao et al., 2022)

Despite their practical value, classical planners struggle in unknown, unstructured, and dynamic environments. This motivates the need for DRL approaches that learn navigation policies from sensor data and reduce the reliance on predefined maps and decision-making rules (Zeng et al., 2021; Zhu et al., 2025)

2.2 Reinforcement Learning Foundations

Reinforcement Learning (RL) is a type of machine learning where the agent learns through interacting with the environment through a trial-and-error approach, and gets scalar feedback known as rewards or penalties (Sutton & Barto, 2018). The RL framework typically consists of an agent that observes a state, interacts with the environment, receives scalar feedback as reward for an optimal action or penalty for a non-optimal action, and updates the policy to maximise the cumulative reward.

In RL, policies are learned rather than predefined rules, which enhances the adaptability of RL to cope with uncertain inputs. The DRL methods are commonly divided into value-based methods and policy-based methods or actor-critic methods (Zhu & Zhang, 2021). Value-based methods such as Deep Q-Network (DQN), Double DQN, Dueling DQN, and Dueling Double DQN (D3QN) learn value functions and select actions using an argmax rule over the learned Q-values. These methods are applicable for discrete action spaces, but they are weaker for continuous action spaces (Zhu & Zhang, 2021). While the policy-based and actor-critic methods, such as Asynchronous Advantage Actor-Critic (A3C), Deep Deterministic Policy Gradient (DDPG), Twin Delayed DDPG (TD3), Proximal Policy Optimization (PPO), and Soft Actor-Critic (SAC), focus on decision-making (Le et al., 2024; Zhu et al., 2025).

To enhance learning in high-dimensional spaces, deep neural networks are integrated into reinforcement learning. Methods like DQN achieve human-level performance in Atari games, where a deep convolutional neural network estimated the value functions from pixel-level inputs (Mnih et al., 2016). Despite its effectiveness, the dependence on a discrete action space

limited its performance and application in continuous control tasks. Lillicrap et al. (2016) introduced a baseline algorithm for continuous control, Deep Deterministic Policy Gradient (DDPG). DDPG is an actor-critic, model-free algorithm that can learn policies in a continuous action space.

Pure RL has issues like slow convergence, low sample efficiency, and instability. Some studies focused on stabilizing the RL through policy optimization techniques (Schulman et al., 2017), entropy regularization (Haarnoja et al., 2018), and addressing function approximation errors (Fujimoto et al., 2018). Methods like Proximal Policy Optimization (PPO), Asynchronous Advantage Actor-Critic (A3C), Soft Actor-Critic (SAC), and Twin Delayed DDPG (TD3) also contributed to the field of robot navigation, are adapted to complex environments, and therefore become central to the DRL-based navigation.

2.3 Deep Reinforcement Learning for Continuous Control

Deep Reinforcement Learning (DRL) is an extension of RL where deep neural networks are integrated to estimate the policy and value functions for continuous action spaces (Lillicrap et al., 2016). The navigation performance in DRL is strongly influenced by the core components of the MDP framework, such as the state representation, the action space, and the reward design. The state representation may include sensor-level states (LiDAR ranges, RGB observations, or depth images) or agent-level states (positions, velocities, and relative distances). The action space can be discrete or continuous velocity commands, and the reward function is often designed with a balance between goal reaching and safety measures. (Zhu & Zhang, 2021; Le et al., 2024).

The deterministic policy gradient provides the theoretical basis for actor-critic methods, enabling optimization of parameterized policies in continuous action spaces. The deterministic policies are suitable for high-dimensional continuous control problems as they reduce the variance in gradient estimates by integrating over the state distribution (Sutton & Barto, 2018). Also, the vision-based approaches conducted by Levine et al. (2016) have proven that the convolutional neural networks could learn visuomotor policies from raw pixel observations and map to motor commands without handcrafted features. Asynchronous

variants accelerated the training by applying a change in the traditional robotics pipeline and considered planning, perception, and control in parallel (Mnih et al., 2016).

However, continuous control remains challenging as they include non-stationary targets, due to bootstrapping-based updates. Additionally, overestimation bias caused by the function approximation errors, and finally, dead policy issues caused by suboptimal deterministic actions by the agent due to inefficient exploration or critic inaccuracies. In robotic settings, where the agent processes high-dimensional sensory inputs through LiDAR or camera images, these challenges escalated, as they are under conditions like environmental uncertainty and partial observability. This shows the necessity of algorithmic enhancements to achieve reliable and robust performance which are discussed in the following section.

2.4 DDPG, TD3, SAC and PPO

Lillicrap et al. (2016) introduced a baseline algorithm for continuous control, Deep Deterministic Policy Gradient (DDPG). DDPG is an actor-critic, model-free algorithm that can learn policies in a continuous action space. It utilizes replay buffers, ensuring sample efficiency and target networks for training stability. Therefore, DDPG is applicable in cases where smooth linear and angular velocities are required, like in mobile robot navigation (Lillicrap et al., 2016).

To illustrate the capability of deep reinforcement learning in continuous action spaces, DDPG was shown to solve more than 20 simulated physics tasks (Lillicrap et al., 2016). But the method has many limitations, such as overestimation bias, hyperparameter sensitivity, exploration noise sensitivity, and reward design dependency. These constraints are significant as they can cause performance degradation (Lillicrap et al., 2016; Zhu et al., 2025).
These weaknesses are addressed by Fujimoto et al. (2018) with Twin Delayed DDPG (TD3). The method introduced three changes:

- Clipped Double-Q learning: to mitigate the overestimation, used a minimum of two critics
- Delayed policy updates: to stabilize the training, the actor is updated less frequently

- Target policy smoothing: for better exploration and to increase the robustness by adding noise.

Thus, TD3 becomes a robust baseline for robotic tasks, as these mechanisms significantly improved the performance on MuJoCo benchmarks.

Soft Actor-Critic (SAC) is another continuous control algorithm that improved the efficiency of off-policy methods through entropy regularization. This encourages exploration and makes SAC more effective in sparse-reward settings (Haarnoja et al., 2018). It adds stochasticity to the policy objective. It outperforms TD3 in sample efficiency and convergent performance in continuous control. Also, its stochastic actor is good for navigation in uncertain environments, as the exploration encourages collision avoidance.

In contrast, Proximal Policy Optimisation (PPO) adopts an on-policy method, prioritizes monotonic improvement and stable updates through clipped optimization by preventing destructive policy updates (Schulman et al., 2017). PPO is popular in robotics due to its simplicity and reliability. It balances the stability with strong results in locomotion and manipulation tasks.

Table 1. Summary of DDPG, PPO, TD3, and SAC for mobile robot navigation.

Algorithm	**Sample Efficiency**	**Stability**	**Performance in Sparse Rewards**	**Standard Role**
DDPG	Medium	Low	Poor	Baseline continuous control
PPO	Low-Medium	Very high	Medium	Stable on-policy control
TD3	Medium-High	High	Medium	Robust baseline for robots
SAC	Medium-High	High	Excellent	Sample-efficient, stochastic

Overall, these algorithms form the algorithmic core for DRL navigation. The deterministic off-policy methods, DDPG and TD3, and the stochastic off-policy method, SAC, provide efficiency and control, while the stochastic on-policy method, PPO, provides robustness and stability. Their application to robot control, especially in dynamic environments, is explored next.

2.5 DRL-Based Robot Navigation

Deep Reinforcement Learning has renewed mobile robot navigation by enabling end-to-end policies that map raw sensor inputs (LiDAR, RGB images, and depth) directly to continuous velocity commands, avoiding explicit mapping or planning (Tai et al., 2017).

A primary task for DRL- based navigation includes local obstacle avoidance without explicit maps, where the robot relies on sensor-level perception states and reaches its goal by avoiding the obstacles, mostly in a simplified environment (Zhu & Zhang, 2021; Le et al., 2024). On average, most implementations use high-dimensional state space (LiDAR or depth image data), continuous or discrete actions, and dense reward shaping for efficient and safe navigation. Also, many studies implement domain randomization or expert trajectories to improve learning and transferability (Zhu & Zhang, 2021; Shakerimov et al., 2023).

Indoor navigation tasks include evaluating the performance of the robot in rooms, corridors, or maze-like layouts. While visual navigation methods utilize RGB inputs, recurrent networks are used to address partial observability. These methods are successful in simulation where the robot must reach a goal based on the perception inputs (Le et al., 2024; Zhu et al., 2025). Generalization has improved with the vision-based and off-policy variants. Gandhi et al. (2017) demonstrated learning from failure, enabling policy acquisition for drone navigation through crash-tolerant exploration. The comprehensive survey, like Zhu et al. (2025) has confirmed the DRL's dominance in mapless environments, even though the performance degrades in highly dynamic crowds without explicit safety constraints.

Multi-Robot Navigation and Multi-Agent Reinforcement Learning (MARL) is an integration of multiple autonomous systems for navigation within a shared environment. Compared to the traditional methods, MARL helps robots to learn together and make decisions in a complex environment with other robots and obstacles. These methods are extremely applicable for robots within a shared space (Zhu et al., 2025). But extra challenges are introduced by MARL, such as training instability and scalability issues with the increase of agents. Besides, most studies still assume simplified settings rather than crowd-rich environments for

experimentation (Zhu et al., 2025; Le et al., 2024). This demonstrates the importance of reward engineering in DRL navigation as it is vulnerable to unsafe exploration.

2.6 Safety-Aware Reward Design

Reward shaping is demanded by DRL navigation for safe navigation with collision-free behaviour while maximising task completion (Miranda et al., 2024). Traditional sparse rewards lead to inefficient learning, while shaped rewards that include potential-based functions, yielding shaped rewards, preserve policy invariance (Chahoud et al., 2025). To improve performance in unknown environments, dense reward shaping is effective as it improves the learning stability, along with obstacle avoidance, goal progress, and control efficiency. But reward shaping alone does not improve the generalization, as most training is performed in narrow environmental distributions. Therefore, a clear need of environment diversity should be considered while training to achieve more robust navigation (Miranda et al., 2024; Le et al., 2024).

As mentioned, one of the major limitations in navigation tasks includes unguaranteed safety from standard DRL-based navigation, as they only promise higher expected return. This is unacceptable in a socially dynamic environment. Safe Reinforcement Learning (SRL) addresses this problem by introducing explicit safety constraints through reward designs, cost functions, or safety critics (Zhou et al., 2023). Their approach has surpassed the success rate of vanilla DDPG by 40% among moving agents. Miranda et al. (2024) improved the generalization by combining curiosity rewards with safety terms that enable zero-shot transfer across unseen environments.

Notably, Conflict-Averse Safe Reinforcement Learning (CASRL) was especially designed for pedestrian-rich navigation. CASRL breaks down the navigation task into a goal-reaching critic and a collision avoidance safety critic with separate policy gradients. This significantly improves the navigation performance in dynamic environments (Zhou et al., 2023). CASRL is important as it demonstrates that safety-aware navigation is achievable in dynamic environments. Also, it shows that rather than terminal collision penalties, the safety can be explicitly treated as a learning objective. Despite improvements, hand-crafted reward shaping

remains fragile, especially in unseen environments. This motivates the need for human/crowd-aware navigation.

2.7 Dynamic and Social Navigation

Dynamic environments are considered one of the major challenges in mobile robot navigation, as the agent should adapt to moving obstacles and environmental variations. These scenarios demand real-time decision-making under uncertainty. This scenario becomes more complex when the agent is deployed in a crowded environment where the agent must avoid collisions by predicting human actions while maintaining human comfort and safety (Rudenko et al., 2020).

In deep reinforcement learning approaches, it learns policies directly from interacting with the environment and does not entirely depend on predefined rules, making it attractive in these settings. Everett et al. (2018) demonstrated the applicability of DRL for motion planning in dynamic environments with other decision-making agents, where the robot must predict the actions of other agents and make decisions adaptable to them. This is an important step in a crowded setting with humans or other agents. In parallel, a safe reinforcement learning approach for autonomous navigation in dynamic environments is proposed by Zeng et al. (2021), highlighting safety constraints and collision avoidance during learning and deployment.

Another concern in social navigation is that the robot must understand the positions of other obstacles and how they are likely to move. This reveals the importance of trajectory prediction and crowd modelling in DRL-based navigation systems. Godoy et al. (2016) focused on human-aware navigation policies for socially acceptable behaviour, while the role of human motion prediction to enable greater interaction-aware planning was highlighted by Rudenko et al. (2020). This showcases the importance of reward design in such settings, as the agent must learn to choose safe and socially acceptable behaviour rather than fast actions.

Another challenge for safe navigation arises in multi-robot and dense crowd systems, because an agent's actions affect the future behaviour of other agents. CrowdMove, a mapless multi-robot navigation framework, was introduced to address these difficulties (Fan et al., 2018). It learns decentralized behaviour from experience. This direction was extended by Le et al. (2024) by studying the multi-robot navigation with DRL. It demonstrated the effectiveness of collaborative learning and communication-aware policies in improving the stability.

Overall, the research on social and dynamic navigation revealed that successful robot motion in crowded dynamic environments requires more than obstacle avoidance. It requires prediction, safety awareness, and socially compliant behaviour. DRL is promising in mobile robot navigation, but its success heavily depends on safety-aware reward shaping, training diversity, and reliable sim-to-real transfer.

2.8 Sim-to-real transfer challenges

One of the major challenges in DRL navigation is the transferability problem. The policies are trained in a simulation with limited setup, and a significant performance degradation can be seen when it is implemented in real time, where it faces different dynamics in sensor noise, hardware constraints, and environmental changes. In case of dynamic scenarios, this gap can lead to unsafe collisions due to errors in perception or control systems (Peng et al., 2018).

Methods like domain randomization, domain adaptation, curriculum learning, and uncertainty modelling are explored to improve the robustness and reduce the sim-to-real gap, but they do not fully mitigate the problem, as the environment is unpredictable in real-time. Peng et al. (2018) demonstrated that domain randomization can support the effective transfer of robotic control task policies, as it trains the policy on different possible environment variations.

In domain adaptation, the variance between the simulation and reality is explicitly reduced by aligning feature distributions or adjusting policy behaviour after deployment. Combining the domain randomization and domain adaptation demonstrated that the two methods can

balance to improve the transfer efficiency (Shakerimov et al., 2023). Since it can be applied without retraining, the adaptation will help the policy to adjust to different floor settings, obstacle shapes, or action delays. This is useful in navigation tasks because small variations in sensor input or robot motion can cause significant performance degradation.

Another method is curriculum learning, which introduces the complexity gradually. Sigal et al. (2024) introduced the curriculum learning for robot navigation, initially from simple scenarios to more complex ones. This caused a significant improvement in stabilized training and generalization. The sim-to-real transfer challenges in dynamic navigation is further discussed in Chukwurah et al. (2024), revealing that the problem is not only due to the visual variations but also about the environmental interaction dynamics, reward sensitivity, and timing issues. Another approach is uncertainty-aware methods. Kahn et al. (2018) showed the importance of considering uncertainty in improving robustness in robot navigation to avoid presuming decisions in unfamiliar environments. In modern robotics, these methods are combined with simulation platforms like ROS2, Gazebo, or PyBullet to build a testing pipeline before real-world deployment (Macenski et al., 2022).

Overall, sim-to-real transfer is one of the major barriers in DRL-based mobile robot navigation. Approaches like domain randomization, domain adaptation, curriculum learning, and uncertainty modelling help to reduce the sim to real gap but not fully eliminate it. Therefore, robust transfer in dynamic environments is still an active research problem, where all interaction patterns are difficult to model in simulation.

2.9 Research Gaps and Summary

Gap 1: Lack of Behavioural Failure Analysis

A major limitation in the existing literature is that most studies evaluate the DRL navigation performance using only quantitative analysis rather than qualitative analysis. They focus on success rate, collision rate, and other quantitative metrics. These metrics are useful, but they do not explain why a policy fails or reveal the recurrent failure patterns.

Table 2. Overview of selected navigation methods, environment, and evaluation criteria.

Study	**Environment type**	**Metrics used**	**Behavioural analysis**
Tai et al. (2017)	Static + real-world	Success rate	Limited
Everett et al. (2018, 2021)	Crowd/dynamic	Collision avoidance, success	Limited
Xie et al. (2021)	Dynamic crowds	Safety, success	Limited
Kahn et al. (2018)	Real-world	Uncertainty, safety	Limited
Fan et al. (2018)	Multi-robot crowd	Success, efficiency	Limited
This work	Static + dynamic	Success rate, collision rate, path length, average reward	Included

For safety-critical navigation, the absence of failure analysis is complicated because it will prevent the researchers from identifying behavioural patterns that lead to unsafe actions. This thesis addresses that gap by conducting a failure behaviour analysis along with quantitative metrics.

Gap2: Limited Systematic Analysis of Safety-aware reward design

Another main issue in the literature is the reactive reward designs that penalize collision only after physical contact occurs. In most cases of DRL navigation systems, the reward is dominated by the goal distance and a large negative terminal penalty for collisions. This reward design is simple, but it does not provide any safety feedback, therefore leading to sparse gradients, slow learning, and unsafe collision-based explorations. Several studies demonstrated the importance of proactive reward shaping in improving safety and convergence by rewarding safe proximity, smoother motion, or obstacle-awareness (Zhou et al., 2023; Miranda et al., 2024). This thesis incorporates systematic reward designing along with proximity-based penalties and smoothness incentives to support more stable continuous control learning.

Gap3: Absence of systematic analysis across different settings

Finally, in the current DRL navigation literature, policy evaluation is conducted in single, isolated environments rather than across various controlled complexity levels. This limits the analysis of the policy's robustness and generalization. For example, Tai et al. (2017) evaluated in a static maze, while Everett et al. (2018) used a fixed number of agent crowds. This makes it difficult to analyze the performance and robustness of the trained agent as the environment complexity evolves.

This thesis addresses this gap by implementing a curriculum-learning framework that systematically increases the difficulty of navigation tasks on each level during training. The curriculum consists of six stages:

- Stage 1: 10×10 m arena, 1 dynamic obstacle, and 2 static obstacles.
- Stage 2: 20×20 m arena, 2 dynamic obstacles, and 2 static obstacles.
- Stage 3: 30×30 m arena, 3 dynamic obstacles, and 2 static obstacles.
- Stage 4: 40×40 m arena, 3 dynamic obstacles, and 3 static obstacles.
- Stage 5: 50×50 m arena, 4 dynamic obstacles, and 3 static obstacles.
- Stage 6: 50x50 m arena, 5 dynamic obstacles, and 4 static obstacles.

This progressive approach allows the policy to learn basic obstacle avoidance and robust navigation skills as the number of obstacles and spatial complexity increase.

Additionally, after the curriculum training, the policy is evaluated under unseen environments without retraining, enabling a generalization on zero-shot, warm-up fine-tuning, and out-of-distribution (OOD) stress sweep assessment modes.

3 Methodology

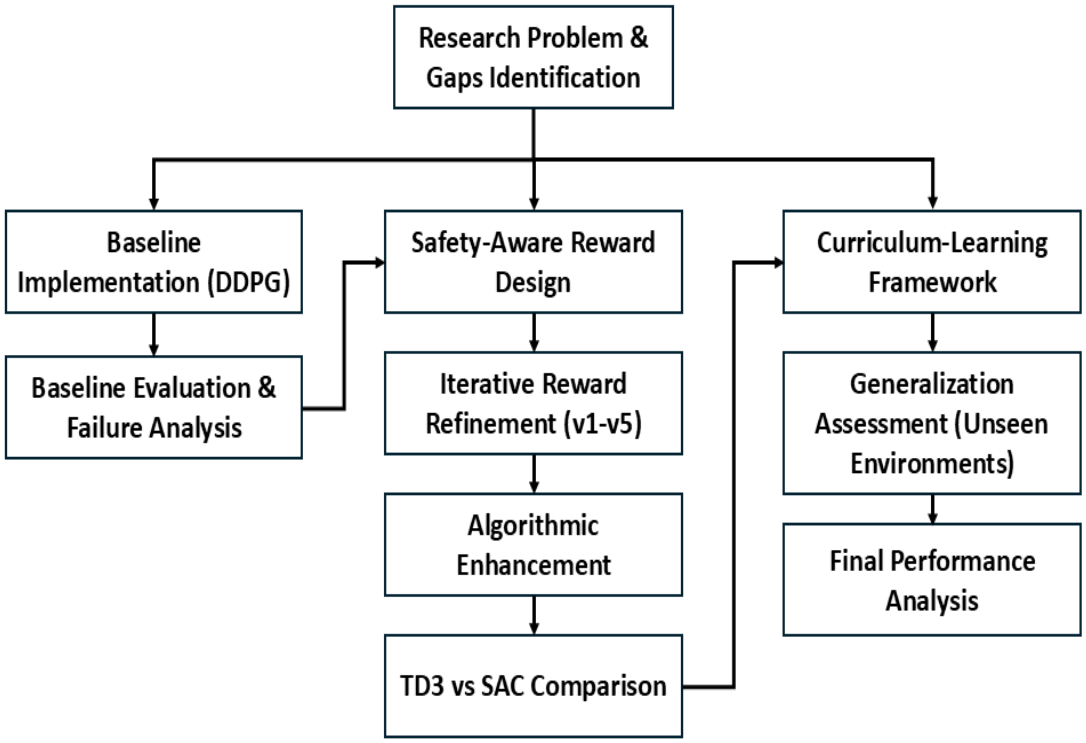


Figure 1. Methodology Flowchart of the research process.

3.1 Baseline Methodology

3.1.1 Simulation environment

A baseline reinforcement learning (RL) framework was developed to evaluate the performance of a standard model-free, continuous-control navigation in a constrained robotic environment. The simulation is implemented using PyBullet, following a Gymnasium-compatible interface. The environment consists of a spherical robot initialized at position (1,1) navigating in a bounded 5 x 5 m 2D arena bounded by four static walls and a fixed target at (4, 4).

The robot's action space was continuous, consisting of linear velocity and angular velocity in x and y directions, where each action lies in the range (-1,1):

$$a_t = [f_x, f_y]$$

The observation space consists of 24-ray LiDAR distance measurements, the robot's 2D position and velocity, yielding a 28-dimensional state representation. This produced a continuous state representation, providing perception for collision avoidance and motion information. The custom environment developed in this thesis is referred to as "SphereNav".

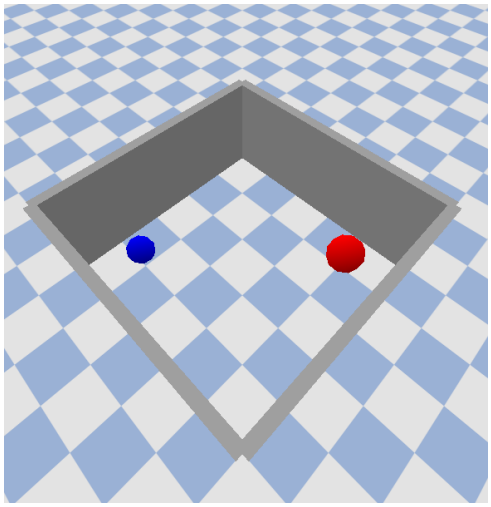


Figure 2. SphereNav simulation environment.

3.1.2 Reward structure

A non-safety-aware reference scenario is established initially for the baseline reward function. For each step, the agent receives a reward:

$$r_t = -0.1 \cdot d_t$$

where d_t is the distance from the robot to the goal at time t.

Besides, the episode terminates:

- If the robot reaches the goal ($d_t < 0.3$), a reward of $+100$ is added for reaching the goal.
- If the robot collides with the wall, a penalty of -100 is applied.

Also, the episode is truncated after a maximum of 500 steps. This setup creates a simple baseline without curriculum learning, guidance, or safety-aware reward design.

3.1.3 Training setup

Deep Deterministic Policy Gradient (DDPG), a model-free, off-policy, actor-critic algorithm for continuous control, is used for training the baseline agent. The policy is trained for 50,000 total timesteps, where each episode is limited to a maximum of 500 steps. A learning rate of $1x10^{-3}$ was considered. Also, to encourage exploration, a Gaussian action noise with a standard deviation (σ) of 0.1 is added. This configuration is developed as a standard baseline where later modifications and improvements could be compared.

3.1.4 Behavioural Evaluation Protocol

The learning behaviour of the agent is evaluated across 100 independent episodes. The evaluation metrics used included success rate, collision rate, average episode return, and average episode length.

In addition, to observe the learning dynamics over time, the TensorBoard logs, including the mean episode reward and mean episode length, were analysed. Also, the qualitative behavioural analysis was performed by tracing agent trajectories and recording the failure behaviour via interaction recording.

3.2 Safety-Aware Reward Design

3.2.1 Overview

This section demonstrates the methodology for Gap 2, focusing on improving the performance of the Deep Deterministic Policy Gradient (DDPG) algorithm. The primary goal is to address the limitations observed in the baseline model in Gap 1, particularly the instability, poor convergence, goal-irrelevant behaviour, and high collision navigation.

Several behavioral failures, such as freezing policies, higher collisions, unstable exploration, and aggressive goal-reaching, were shown in early reward design experiments. These limitations motivated the expansion of the safety-aware reward system that balances goal progression, stable exploration, and safe navigation behavior.

3.2.2 Refined Safety-Aware Reward Design

To address the higher goal-crashing behavior and unstable exploration during preliminary experiments, implement a capped progress reward along with dense multi-component shaping and proportional safety penalties to enable long, safe exploration with precise termination.

Reward function:

$$r_t = r_{progress} + r_{goal} + r_{align} - r_{action} + r_{safety} + r_{terminal}$$

where:

- $r_{progress}$ = 20.0 * clip (Δd_{goal}, 0, 0.5), capped sparse progress
- $r_{goal} = \frac{1}{d_{goal}+1}$, dense goal shaping
- $r_{align} = 3.0 * clip(vec\{v\}_t * \{g\}, -0.5, 1.0)$, velocity-goal alignment
- $r_{action} = 0.0005|a_t|_2$, minimal action regularization
- $r_{safety} = \begin{cases} -5.0 \cdot (0.6 - \min(lidar_t)), & \min(lidar_t) < 0.6 \\ 0, & \text{otherwise} \end{cases}$
- $r_{terminal} = \begin{cases} +100, & \mathrm{d_{goal}} < 0.45 \\ -15.0, & \text{collision} \\ -5.0, & \mathrm{t} \geq 1000 \end{cases}$

Implementation and parameter tuning: The observation space extended to 31 dimensions, including 24-ray lidar, relative goal position, robot position, robot velocity, and goal distance. Improved the starting position distribution $p_o \sim U[0.8,1.8]$ [2] and max steps to 1000. Robot downsized to r=0.25m, and the arena inset to 4.8m to provide natural safety margins.

3.2.3 Final Reward Design with TD3 and SAC

The previous versions clearly justified the overestimation and inefficient exploration of the DDPG algorithm. The limitations observed during preliminary experiments motivated the algorithm advancement from DDPG to TD3 (Twin Delayed DDPG) and SAC (Soft Actor-Critic). TD3 eliminates the overestimation through clipped double Q-learning and delayed policy updates. While SAC integrates entropy regularization for exploration. For fair comparison, the same environmental setup, reward design, and hyperparameters are used.

Reward design:

$$r_t = r_{progress} + r_{time} + r_{control} + r_{safety} + r_{goal-shape} + r_{terminal}$$

where:

- $r_{progress} = 8.0(d_{t-1} - d_t)$, is the progress reward toward the goal
- $r_{time} = -0.02$, is the per-step time penalty
- $r_{control} = -0.01\|a_t\|^2$, is the action regularization
- $r_{safety} = \begin{cases} -4.0 \cdot \left(\frac{0.55 - min(lidar_t)}{0.55}\right)^2, & \min(lidar_t) < 0.55 \\ 0, & \text{otherwise} \end{cases}$
- $r_{goal-shape} = 0.3\left(1 - tanh(d_t)\right)$, is the long-range pull to the goal.
- $r_{terminal} = \begin{cases} +150, & \mathrm{d_{goal}} < 0.45 \text{ (success)} \\ -120, & \text{collision} \\ 0, & \text{otherwise} \end{cases}$

Implementation: The arena size is 6m x 6m with a 0.2 m safety margin. The observation space is 31 dimensions, including 24-ray LIDAR, relative goal position, robot position, robot velocity, and goal distance. The action $v_x, v_y \in [-1,1]$ is scaled x 4.0 internally. Both algorithms, TD3 and SAC, are used for training.

Parameter tuning

The parameter tuning for both algorithms is shown in Table 3.

Table 3. Parameter tuning values for TD3 and SAC

Hyperparameter	**TD3**	**SAC**
Learning Rate	3e-4	3e-4
Buffer Size	1,000,000	1,000,000
Batch Size	128	256
Policy Network	ReLU	ReLU
Discount factor (γ)	0.99	0.99
Soft update (τ)	0.005	0.005
TD3 policy delay	2	-
SAC entropy	-	“auto”

3.3 Algorithm Selection and Pre-Curriculum Evaluation

3.3.1 Controlled training

This section demonstrates the methodology for selecting the reinforcement learning algorithm prior to introducing curriculum training. Comparing the performance evaluation of Twin Delayed DDPG(TD3) and Soft Actor-Critic (SAC) under controlled settings.

The earlier experiments in the were performed in a simplified navigation setting, with random starting positions and a fixed target. For the goal-reaching task, both algorithms achieved 100% success and 0% collisions, but without obstacles. These conditions motivated this comparative evaluation.

To differentiate their robustness, a challenging environmental evaluation was required. Therefore, two environments with static obstacles and dynamic obstacles were introduced. The objective of this evaluation is to select an algorithm for curriculum learning.

The reward function used in this stage was changed compared to the earlier reward function to support navigation under an environment containing obstacles. This revised reward structure helped for stable exploration, smooth navigation behavior, collision avoidance, and task completion with safe termination in both static and dynamic environments.

3.3.2 Simulation environment

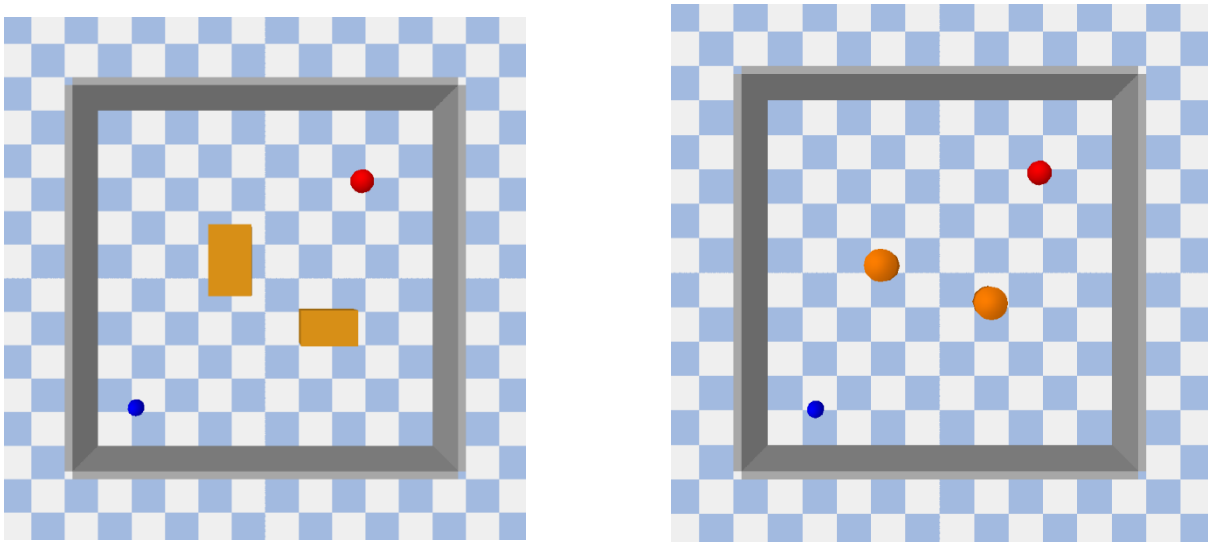
To evaluate robustness under obstacle-aware navigation, two controlled environments were created using PyBullet:
Static Obstacle Environment

- Arena size 10 x 10 m
- Four boundary walls
- Two fixed static obstacles
- Spherical robot agent of radius 0.25m
- Fixed goal position at (7.8, 7.8)

The observation space was 31 dimensions, including 24-ray LiDAR, relative goal position, robot velocity, robot position, and goal distance.

Dynamic Obstacle Environment

- Identical environment as static but extended by adding time-dependent obstacle motion.
- Stochastic angular velocity updates (ω ∈ [0.35, 0.95])
- Non-stationary environment dynamics



(a) Static environment with 2 obstacles (b) Dynamic environment with 2 obstacles

Figure 3. Static and dynamic environment with two obstacles

Reward function design

The reward design is same for both static and dynamic environment across SAC and TD3.

Reward design:

$$r_t = 12.0(\Delta d_{\text{goal}}) - 0.015 - 0.01\|a_t\|^2 + r_{safety} + r_{align} + r_{slow} + r_{stall} + r_{terminal}$$

where:

- $12.0(\Delta d_{\text{goal}})$ is strong progress shaping
- -0.015 is a per-step time penalty to encourage faster completion
- $-0.01\|a_t\|^2$ is action regularization
- $r_{safety} = \begin{cases} -8.0 \cdot \left(\frac{0.9 - d_{\min}}{0.9}\right)^2, & d_{\min} < 0.9 \\ 0, & \text{otherwise} \end{cases}$
- $r_{align} = +0.2(\hat{v}_t \cdot \hat{g}_t)$ where:
  - $\hat{v}_t$ is the normalized robot velocity
  - $\hat{g}_t$ is the normalized goal direction

- $+0.2$ is the velocity-goal alignment bonus

- $r_{slow} = \begin{cases} -0.04 \cdot \text{speed} \cdot (1.1 - \min(lidar_t)), & \text{d}_{\min} < 1.1 \\ 0, & \text{otherwise} \end{cases}$
- $r_{stall} = \begin{cases} -0.01, & \text{step distance} < 0.004 \\ 0, & \text{otherwise} \end{cases}$
- $r_{terminal} = \begin{cases} +200, & \text{d}_{\text{goal}} < 0.55 \\ -180, & \text{collision} \\ 0, & \text{otherwise} \end{cases}$

Implementation: Both algorithms are trained using identical environmental dynamics, observation space, and reward function to ensure a fair comparison under controlled settings.

Parameter tuning: The parameter tuning for both algorithms is given in Table 4.

Table 4. Hyperparameters for TD3 and SAC in Controlled Setting

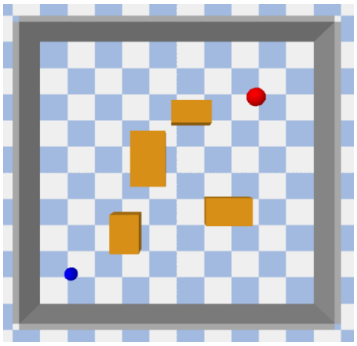
Hyperparameter	**TD3**	**SAC**
Learning Rate	3e-4	3e-4
Buffer Size	1e6	1e6
Batch Size	128	256
Policy Network	ReLU	ReLU
Discount factor (γ)	0.99	0.99
Soft update (τ)	0.005	0.005
TD3 policy delay	2	-
SAC entropy	-	“auto”

3.3.3 Unseen Environment Testing

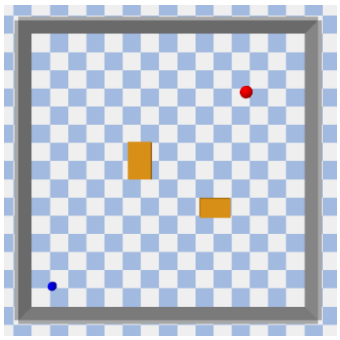
The previous experiments were conducted under controlled settings where there exists a risk that the agent learns memorization rather than navigation methods. This limitation is addressed by performing an additional evaluation to test the generalization performance of both TD3 and SAC in unseen environments without pretraining.

To ensure that the agent cannot depend on memorized trajectories, four unseen scenarios were created to introduce distributional shifts:

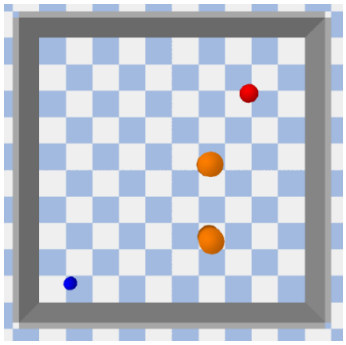
- **UA:** Increased the number of static obstacles from 2 to 4.
- **UB:** Increased the scale of the arena from 10 x 10 m to 15 x 15 m.
- **UC:** Increased the velocity of dynamic obstacles (ω × 2).
- **UD:** Shifted the obstacle positions from the training layout.



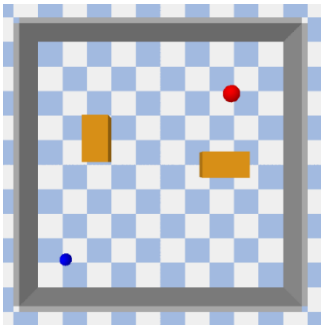
(a) UA: Increased obstacle density



(b) UB: Larger environment scale



(c) UC: Faster dynamic obstacles



(d) UD: Shifted obstacle positions

Figure 4. Four unseen environments for algorithm selection

3.4 Curriculum Training

3.4.1 Overview

This section implements a 6-stage curriculum training with systematic progression of complex environments, including obstacles, arena size, and agent speed. To ensure mastery before advancing to the next stage, stage-gated progression is also enabled.

The reward function used prioritized transferable navigation behavior rather than environment-specific optimization. As the environmental complexity progressively increases during the curriculum-learning stage, there exists a need for further refinement of the reward function. Because the trained policy must balance safety, stability, exploration, and goal-reaching simultaneously.

3.4.2 Curriculum Stages and Environments

For efficient learning, a stage-based curriculum framework is implemented that gradually increases the complexity level. All six predefined stages share the environment developed in PyBullet with:

- Observation space: 31 dimensions (24-ray LiDAR + relative goal position (x, y) + robot velocity (x, y) + goal direction (x, y) + goal distance).
- Randomized starting positions for each episode
- Fixed goal position
- Action space: $[-1, 1]^2$ normalized forces
- Dynamics:
    - Spherical robot with radius r=0.25 m
    - Static boxes
    - Dynamic cylinders

The training is divided into six stages, where each stage increases the complexity based on:

- Number of static obstacles
- Number of dynamic obstacles
- Dynamic obstacle velocity
- Arena size

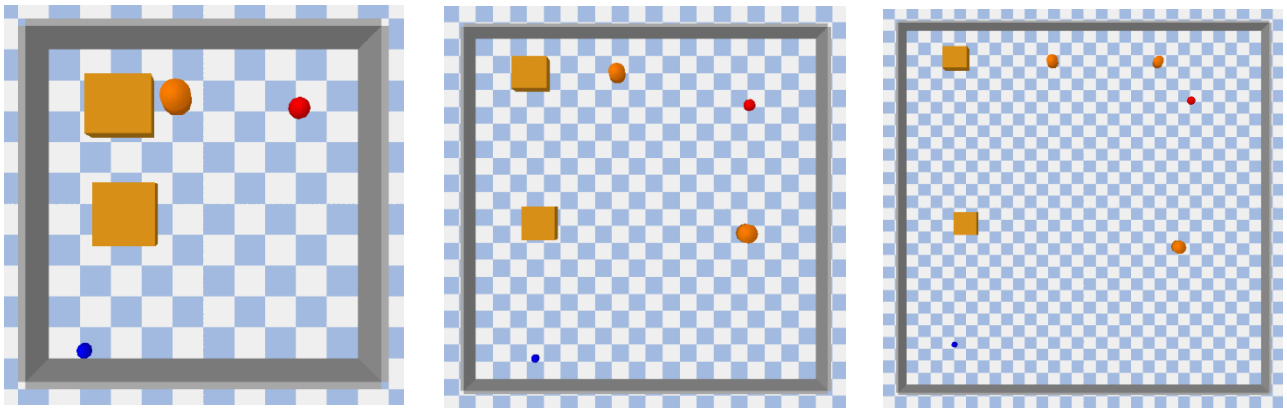
In Stage 6, exactly 50% of dynamic obstacles use orbital motion. In Stages 1–5, all dynamic obstacles use sinusoidal motion.

The environmental progression with gradual scaling to ensure smooth transfer of learned policy across different complex environments is demonstrated in Table 5.

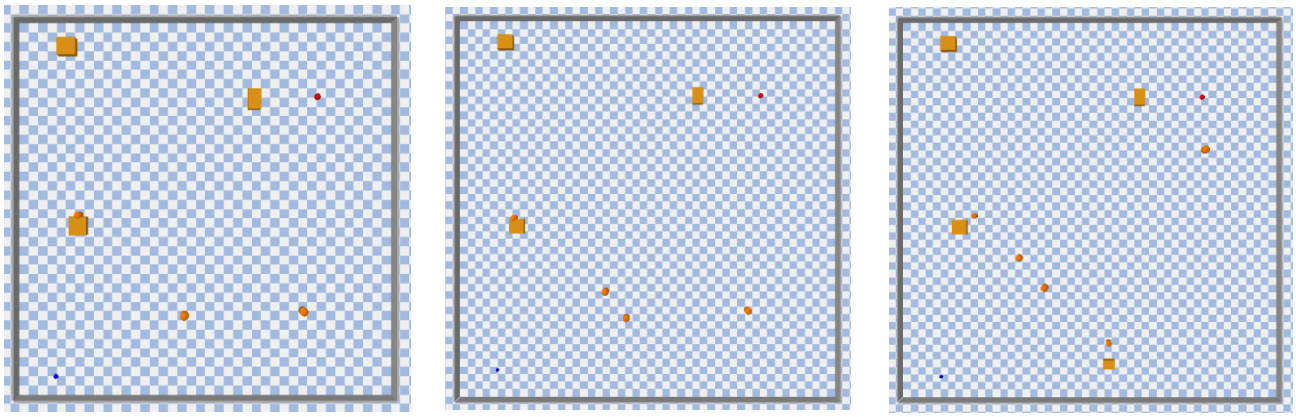
Table 5. Progressive stages of curriculum training

Stage	Static obstacles	Dynamic obstacles	Arena size	Dynamic speed	Timesteps
Stage 1	2	1	10 x 10 m	0.45	300k – 500k
Stage 2	2	2	20 x 20 m	0.65	500k – 800k
Stage 3	2	3	30 x 30 m	0.85	800k – 1.2M

Stage 4	3	3	40 x 40 m	1.00	1.2 – 1.8M
Stage 5	3	4	50 x 50 m	1.20	1.5 – 2.5M
Stage 6	4	5	50 x 50 m	1.40	1.8 – 3.0M



(a) Stage 1: 1 dynamic + 2 static (b) Stage 2: 2 dynamic + 2 static (c) Stage 3: 3 dynamic + 2 static



(d) Stage 4: 3 dynamic + 3 static (e) Stage 5: 4 dynamic + 3 static (f) Stage 6: 5 dynamic + 4 static

Figure 5. Stages in curriculum training (Stage 1-6)

3.4.3 Reward function design

To ensure a balance between safety and goal achievement, a reward function is designed.

Reward design:

$$r_t = 12.0\left(\Delta d_{\text{goal}}\right) - 0.015 - 0.01\|a_t\|^2 + r_{safety} + r_{align} + r_{slow} + r_{goal-shape} + r_{stall} + r_{terminal}$$

where:

- $12.0\left(\Delta d_{\text{goal}}\right)$ is the progress reward towards the goal.
- -0.015 is the per-step time penalty.

- $-0.01\|a_t\|^2$ is the action regularization.
- $r_{safety} = \begin{cases} -8.0 \cdot \left(\frac{0.9-\mathrm{d_{min}}}{0.9}\right)^2, & \mathrm{d_{min}} < 0.9 \\ 0, & \text{otherwise} \end{cases}$
- $r_{align} = 0.2\ (\hat{v} \cdot \hat{g})$, where:
    - $\hat{v}$ = normalized velocity
    - $\hat{g}$= normalized direction to goal
- $r_{slow} = \begin{cases} -0.04 \cdot \|\mathrm{v}\| \cdot \frac{1.1-\mathrm{d_{min}}}{1 \cdot 1}, & \mathrm{d_{min}} < 1.1 \\ 0, & \text{otherwise} \end{cases}$
- $r_{goal-shape} = 0.25(1.0 - \tanh(d_t))$, is the long-range goal attraction.
- $r_{stall} = \begin{cases} -0.01, & \text{step distance} < 0.004 \\ 0, & \text{otherwise} \end{cases}$
- $r_{terminal} = \begin{cases} +220, & \mathrm{d_{goal}} < 55 \\ -180, & \text{collision} \\ 0, & \text{otherwise} \end{cases}$

3.4.4 Stage Transition and Gate Mechanism

A performance-based gating mechanism is enabled between the transition of stages under the following conditions:

- Success rate ≥ 0.75
- Collision rate ≤ 0.15
- Low variance in success rates
- Low variance in episode rewards

The evaluation is conducted over 100 episodes and advances to the next stage only when both performance and stability meet the criteria. This ensures consistent, stable policy behaviour along with performance improvements.

3.4.5 Parameter tuning

Table 6. Hyperparameter settings for curriculum training

Parameter	**Value**
Learning rate	3e-4

Buffer size	1e6
Batch size	256
Discount factor (γ)	0.99
Tau (τ)	0.005
Network	ReLU

Depending on the difficulty level, each stage is trained with a predefined timestep budget ranging from 500k to 2.5M steps. For evaluation (100 episodes) and checkpointing, the training is periodically interrupted every 25,000 steps. Also, to enable recovery and reproducibility, model checkpoints and normalization statistics are saved at regular intervals.

3.5 Generalization assessment

A primary limitation of many DRL policies is their tendency to overfit to the training distribution. To evaluate the generalization capability of the trained reinforcement learning policy, a three-paradigm assignment protocol was implemented. It includes zero-shot transfer testing, warm-up adaptation, and out-of-distribution (OOD) stress testing. These paradigms examined robustness across five environmental dimensions: obstacle density, arena scale, dynamic speed, static obstacles, and motion patterns.

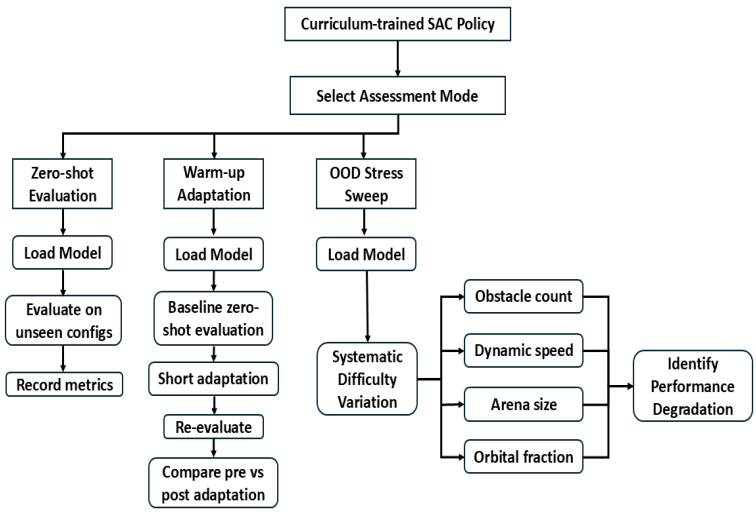


Figure 6. Assessment methodology workflow

3.5.1 Zero-Shot Evaluation

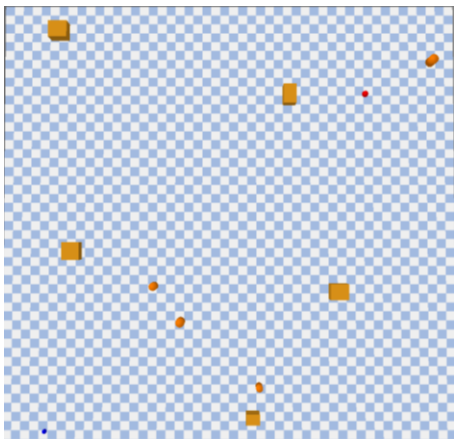
In zero-shot evaluation, the policy was tested directly across 11 unseen configurations that vary in obstacle density, dynamic speed, environment scale, motion patterns, and their combined stages. With a fixed base seed of 99, each configuration is evaluated over 100 episodes. The curriculum training baseline Stage 5 (3 static, 5 dynamic, 50 x 50 m arena, speed = 1.20 m/s, sinusoidal motion) is used as a reference for all OOD deviations.

Table 7. Zero-shot OOD evaluation configurations.

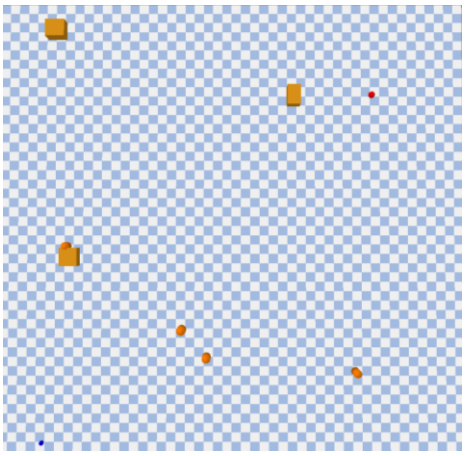
ID	Axis	Static obstacles	Dynamic obstacles	Arena	Speed	Orbital fraction	Description
ZS-D1	Density	5	4	50	1.20	0.0	Higher static density
ZS-D2	Density	3	7	50	1.20	0.0	Higher dynamic density
ZS-D3	Density	5	6	50	1.20	0.0	Mixed high-density
ZS-S1	Speed	3	4	50	1.50	0.0	25% above training
ZS-S2	Speed	3	4	50	2.00	0.0	67% above training
ZS-A1	Arena	2	2	15	0.85	0.0	Unseen smaller arena
ZS-A2	Arena	4	5	70	1.20	0.0	Unseen larger arena
ZS-M1	Motion	3	4	50	1.20	1.0	Fully orbital (unseen motion)
ZS-M2	Motion	3	4	50	1.20	0.5	Mixed orbital/sinusoidal
ZS-H1	Combined	5	6	50	1.60	0.0	Density and speed combined
ZS-H2	Combined	6	8	50	2.00	0.0	Extreme density and speed.

These five generalization axes are selected to examine the robustness of the policy on distinct aspects. The density axis tests the performance of the policy when the obstacle count exceeds the training levels. The speed axis tests whether the policy's safety margin remains sufficient at the training maximum, temporal generalization. The arena axis tests the spatial generalization across smaller (15 x 15 m) and larger (70 x 70 m) environments. The motion

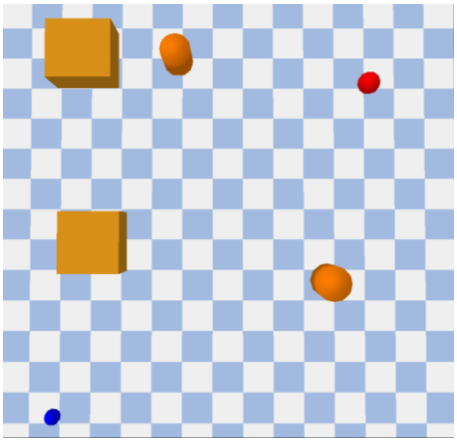
pattern axis tests orbital motion, whereas Stages 1-5 train exclusively on sinusoidal motion. The combined axis increases multiple difficulties, mirroring conditions a deployed policy might encounter in real-world situations.



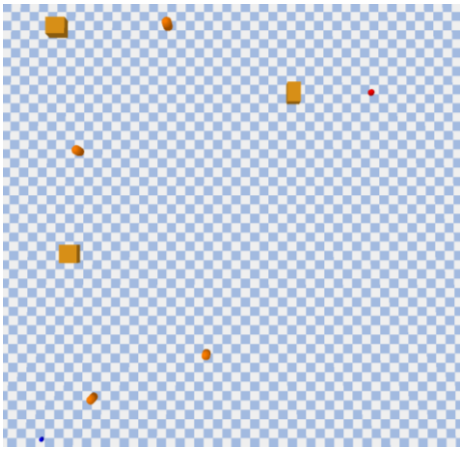
(a) High obstacle density



(b) Faster dynamic obstacles



(c) Smaller unseen arena (15 x 15)



(d) Orbital Motion

Figure 7. Zero-shot Assessment Environments

3.5.2 Warm-Up Fine-Tuning

To evaluate the adaptation capability of the policy, three hard configurations (WU-D: high dynamic density, WU-S: high speed, and WU-H: combined hard), warm-up fine-tuning is performed. The policy was first assessed with the zero-shot performance baseline, followed by a limited adaptation in the target environment for a fixed number of steps (10,000 gradient

steps) and re-evaluated across 100 episodes on a fresh set of seeds. For each configuration, the delta success rate (Δ SR) and delta collision rate (Δ CR) are reported.

3.5.3 OOD Stress Sweep

To analyze robustness and identify performance degradation points, key environment parameters were varied systematically across five individual axes, where each axis is swept independently with other parameters at Stage 5 training values.

Table 8. OOD stress sweep configurations

Axis	Sweep values	Training values	Fixed parameters
Dynamic obstacles	3, 5, 7, 9, 12	4	3 static, 50 x 50 m arena, speed 1.20
Static obstacles	3, 5, 7, 10	3	4 dynamic, 50 x 50 m arena, speed = 1.20
Dynamic speed	1.20, 1.50, 1.80, 2.20, 3.00	1.20	3 static, 4 dynamic, 50 x 50 m arena
Arena size	10, 25, 50, 70, 100	50	3 static, 4 dynamic, speed = 1.20
Orbital fraction	0.0, 0.25, 0.50, 0.75, 1.00	0.0	3 static, 4 dynamic, 50 x 50 m arena, speed = 1.20

The variations included the number of dynamic obstacles from 3 to 12, the static obstacles from 3 to 10, the dynamic speed from 1.20 m/s to 3.00 m/s, the arena size from 10 m to 100, and the orbital fraction from 0% to 100%, transitioning from sinusoidal to fully orbital. Each stress configuration is evaluated for 50 episodes, resulting in 24 total stress configurations.

3.5.4 Evaluation Metrics

Across all three assessment modes, a uniform set of metrics is used. The main evaluation metrics are success rate, collision rate, timeout rate, average episode reward, and average path length. To provide accurate coverage, 95% Wilson score confidence intervals (CI) are used to report all binary proportions. These metrics provide quantitative outcomes along with navigation quality evidence.

The timeout metrics help to diagnose the freezing behaviour of the policy, i.e., a high timeout rate with low collision rate indicates that the policy has learned to avoid obstacles by being still rather than navigating.

3.5.5 Experimental setup

To ensure the validity of the generalization assessment, several experimental controls are applied. First, to prevent leakage of the evaluation episode configuration, all evaluation seeds with base seed = 99 are kept separate from all training seeds with base seed = 42. Second, action selection is fully deterministic (deterministic = True) for all evaluations to isolate policy performance from stochastic action effects. Third, the starting positions are sampled from the same bottom-left region as training, ensuring that performance is analyzed across OOD configurations with obstacle complexity rather than start-goal position geometry changes.

Due to higher computational demands, selected experiments were trained using the GPU partition of the Puhti supercomputer provided by the CSC – IT Center for Science, Finland. Experiments were conducted in the local machine if service interruptions and longer queue delays occurred. Identical hyperparameters and code were used across both platforms to maintain consistency.

4 Results

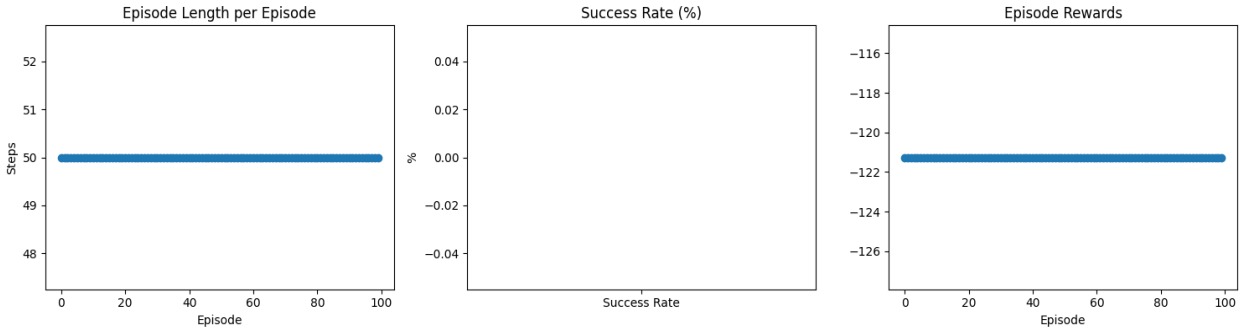
4.1 Behavioural Analysis

4.1.1 Quantitative Performance

The baseline DDPG agent exhibits clear failure across all evaluation metrics as shown in Table 9.

Table 9. Baseline DDPG performance metrics across 100 evaluation episodes.

Metrics	**Baseline**
Success Rate	0.0%
Collison Rate	100%
Average Episode Length	50 steps
Average Episode Reward	-121.26



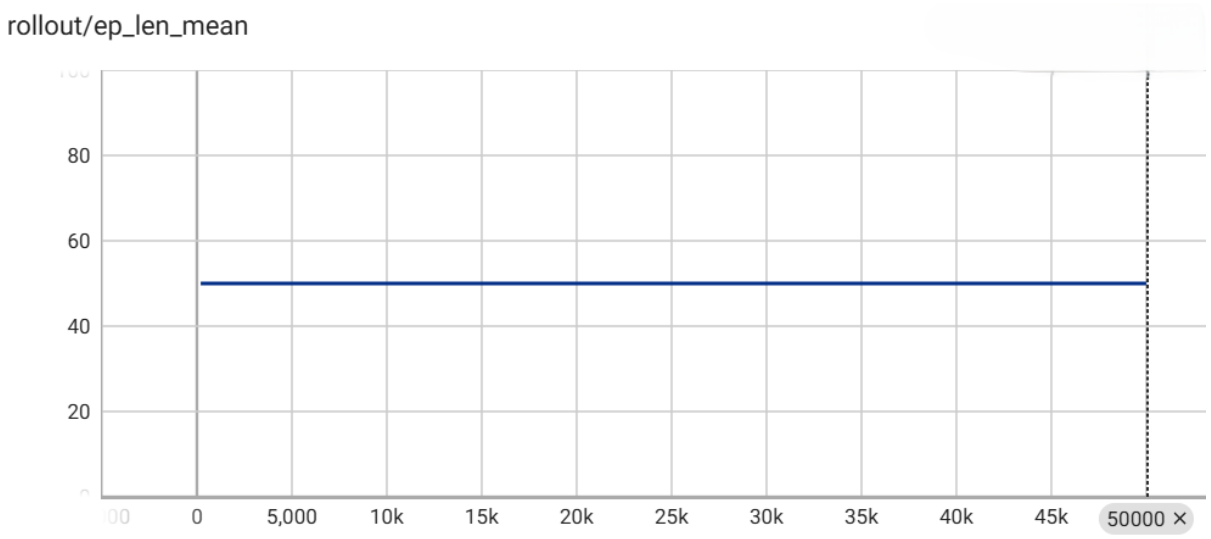
(a)The path length remains constant at 50 steps. (b) 0% success rate (c) The mean reward around −121.

Figure 8. Quantitative performance plots of the baseline DDPG agent

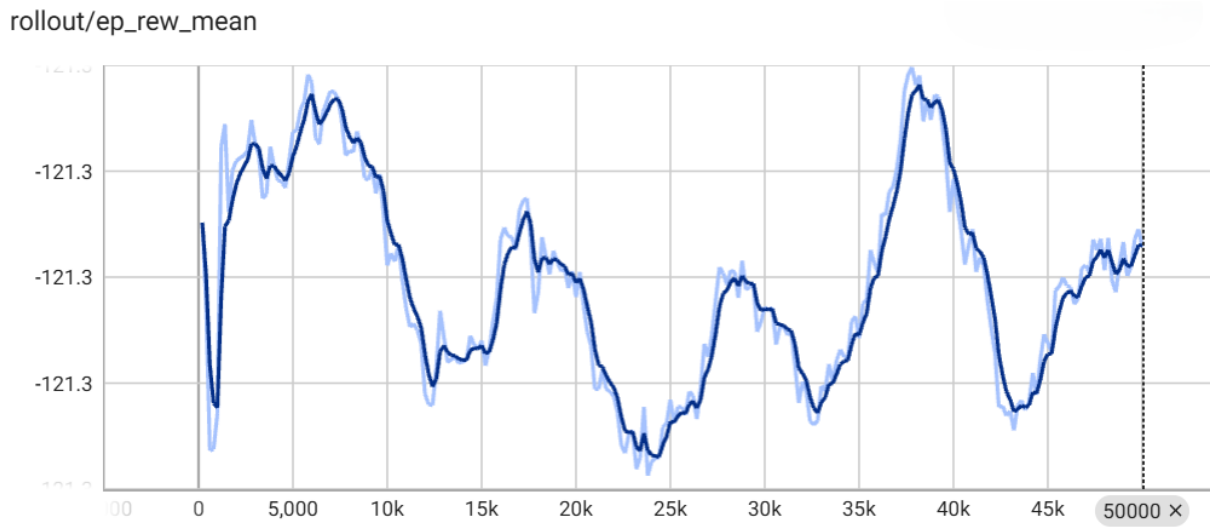
Figure 8 represents the quantitative performance of the baseline over 100 episodes. The episode length remained constant at 50 steps corresponding to early termination. The 0% success rate confirms that the agent did not complete the task. The mean reward stabilized around -121, showing poor performance and a lack of meaningful exploration.

4.1.2 Learning Dynamics

The TensorBoard logs show the failure behaviour of the agent, without a navigation objective. The mean episode length stabilizes at 50 steps throughout the training and is significantly below the maximum episode length of 500 steps. This shows that the agent fails to reach the goal, and the episode is terminated too early due to the agent's collision with the bounding walls. Similarly, the mean episode reward varies and becomes consistent around a negative value of -121.26, showing a lack of meaningful learning.



(a) Episode length stabilizes at 50 steps due to consistent collision termination.



(b) Episode reward (mean ± std) remains at approximately -121 throughout 50,000 timesteps

Figure 9. Baseline DDPG training curves from TensorBoard logs.

This proves that the baseline reward design and the exploration settings alone were insufficient for a stable learning policy in this environment.

For failure behavior analysis, the baseline model DDPG was intentionally designed as a lightweight reference model that was trained for only 500,00 time steps. Therefore, the behavioral analysis should be executed as suggestive characteristics of the baseline rather than final limitations of the DDPG algorithm.

4.1.3 Behavioural Failure Modes

Qualitative analysis of the agent's learning behaviour shows several consistent learning patterns:

- High-collision navigation: Consistent collision with the surroundings, the bounding walls, clearly indicates the lack of an obstacle avoidance strategy.
- Early episode termination: Due to inefficient policy learning and poor exploration, most of the episodes are terminated before reaching the target position.
- Goal-irrelevant navigation: No meaningful trajectories showing the agent's movement towards the goal are produced; instead, unstable trajectories are generated.
- High variant motion behaviour: The agent shows highly unstable movement behavioural patterns under sparse reward conditions

These behaviours are consistent with the known limitations of the vanilla DDPG and are confirmed by simulation recordings, in which the agent fails due to frequent collisions and failure to reach the goal. The consistent wall collisions are documented in the supplementary video. Overall, the baseline DDPG agent fails to learn a stable, goal-oriented policy. This is due to the lack of a safety-aware reward system, a sparse reward structure, and a lack of curriculum learning. These findings clearly establish Gap 1: Lack of behavioural analysis and motivate the improvements explored in the following experiments.

4.2 Safety-Aware Reward Design Evaluation

The second experimental gap demonstrated the transition from complete failure DDPG model to partially competent navigation through reward shaping and safety integration. Earlier reward design experiments revealed unsafe and inefficient navigation under basic DDPG training. These limitations are addressed and improved through algorithm advancement from

DDPG to TD3 and SAC. This experiment achieved a perfect performance of 100% success rate and 0% collision rate.

Table 10 compares the refined safety-aware DDPG baseline with the final TD3 and SAC-based safety-aware framework.

Table 10. Algorithm Performance Comparison

Version	Algorithm	SR	CR	Avg. Reward	Path Length
Refined safety-aware DDPG baseline	DDPG	50.5%	100%	-6262	861 steps
Final Safety-Aware Framework	TD3	100%	0%	135.99	3.79 m
Final Safety-Aware Framework	SAC	100%	0%	136.15	3.79 m

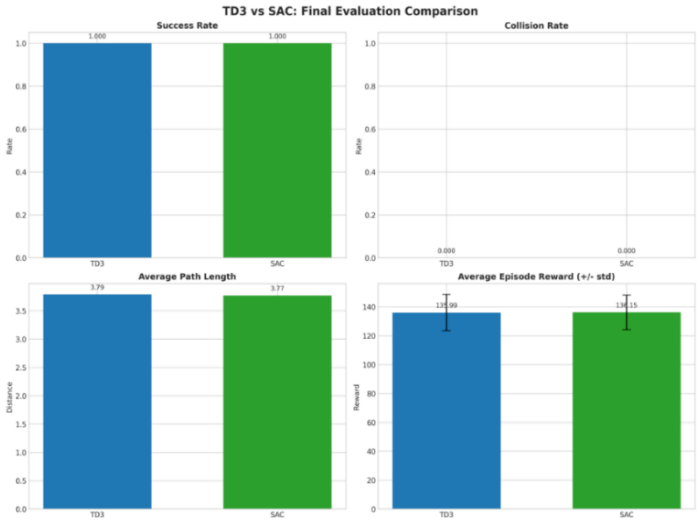


Figure 10. Performance metrics for TD3 and SAC

In the refined version, the evaluation results clearly demonstrate a behavioural paradox. A success rate of 50.5% is attained as the success condition is triggered when the agent reaches the predefined goal radius (0.3m). But the collision rate of 100% is due to the independent evaluation of the collision condition that triggered due to collisions with the wall, and on

reaching the goal. As a result, it clearly demonstrated that the agent gained goal-reaching capability but failed in safe termination behavior due to aggressive goal reaching.

Performance Analysis

- Achieved safety by eliminating higher collision rates through clipped double Q-learning (TD3) and entropy regularization (SAC) that prevents DDPG's Q-value overestimation during final goal approach.
- Path length reduced from 861 steps to 3.79m (direct trajectories)
- Identical performance across TD3 and SAC with (135.99 vs 136.15 reward)shows the algorithm equivalence, where either algorithm succeeds where DDPG struggled.

4.3 Algorithm selection comparative analysis

4.3.1 Controlled Training Results

Static environment

The performance metrics of both algorithms on static environment with 2 obstacles are given in Table 11.

Table 11. Performance metrics on SAC and TD3 in static environment

Metric	**SAC**	**TD3**
Success Rate	100%	100%
Collision Rate	0%	0%
Average Reward	317.25	316.64
Average Path Length	8.92	9.01

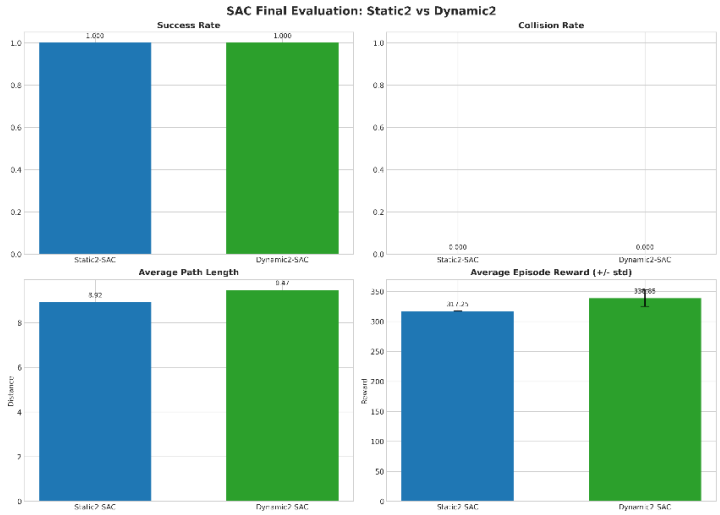
Both the algorithms, TD3 and SAC achieved strong performance of 100% success and 0% collisions in static environment under controlled settings. Even though they achieved nearly identical rewards, SAC obtained a slightly higher average reward (317.25) and shorter path length (8.92m) compared to TD3.

Dynamic environment

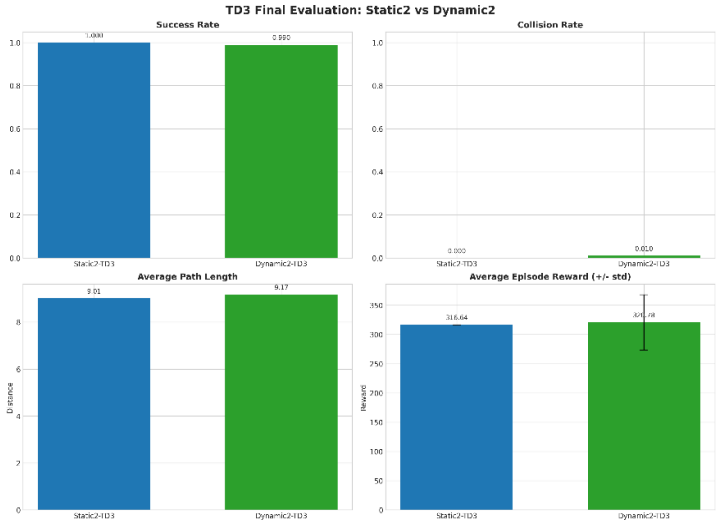
The performance metrics of both algorithms on dynamic environment with 2 obstacles are given in Table 12.

Table 12. Performance metrics on SAC and TD3 in a dynamic environment

Metric	SAC	TD3
Success Rate	100%	100%
Collision Rate	0%	0.01%
Average Reward	338.85	320
Average Path Length	9.47	9.17



a) Performance metrics plot of SAC in static and dynamic environments



b) Performance metrics plot of TD3 in static and dynamic environments

Figure 11. Evaluation metrics plot of SAC and TD3.

The performance gap between the two methods was slightly visible in a dynamic environment. Here, SAC achieved 100% success rate and 0% collisions, and TD3 shows a slight degradation of 0.01% collisions.

4.3.2 Unseen Generalization Evaluation

An unseen evaluation was conducted on a previously trained controlled model to determine the algorithm that is more suitable for curriculum learning. A generalization test on four environments was conducted. The results were demonstrated in Table 13.

Table 13. Performance of SAC and TD3 in unseen evaluation

Test scene	**SAC Success**	**SAC Collision**	**SAC Reward**	**TD3 Success**	**TD3 Collision**	**TD3 Reward**
UA	0%	100%	-175.36	0.00	1.00	-170.25
UB	100%	0%	402.52	0.00	1.00	-42.61
UC	73%	27%	205.09	79%	21%	226.79
UD	100%	0%	330.77	0%	100%	-101.85

SAC achieved a mean success rate of 0.6825, a mean collision rate of 0.3175, with an average reward of 190.77 across all unseen conditions. While TD3 shows a performance degradation with a mean success rate of 0.1975, a mean collision rate of 0.8025, and a negative average reward of -21.98. This clearly indicates that the SAC provides better performance in generalization than TD3 in the given limited scenarios.

Since SAC demonstrated better transfer to larger arenas, dynamic speed changes, and shifted obstacle positions than TD3, SAC is selected for curriculum learning.

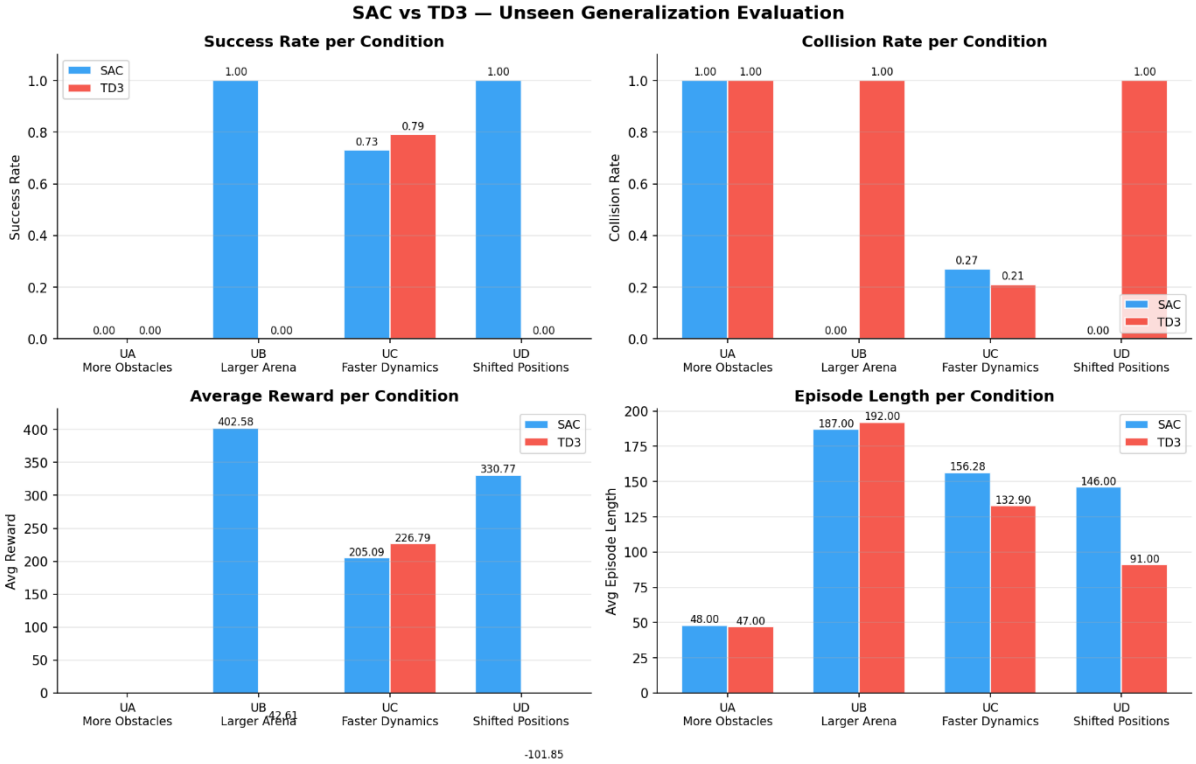


Figure 12. Evaluation metrics plot of SAC and TD3 across four unseen environments

4.4 Curriculum training results

The final evaluation results for all six curriculum stages are evaluated over 100 episodes. The key metrics used for performance evaluation were success rate, collision rate, average episode length, and average rewards.

Table 14. Curriculum stage evaluation results

Stage	Static	Dynamic	Arena	Speed	SR	CR	Avg Length	Avg Reward	Std. Reward
1	2	1	10 x 10	0.45	0.81	0.09	420.8	234.25	416.54
2	2	2	20 x 20	0.65	0.98	0.02	205.4	475.19	66.25
3	2	3	30 x 30	0.85	0.97	0.03	271.1	606.86	95.21
4	3	3	40 x 40	1.00	0.99	0.01	335.0	752.34	74.76
5	3	4	50 x 50	1.20	0.96	0.04	416.8	875.30	174.87
6	4	5	50 x 50	1.40	0.97	0.03	409.8	879.08	137.62

From the metrics in Table 14, Stage 1 performs the worst compared to later stages despite a less complex environment. This happened as the agent had not developed an exploration strategy or stable navigation behavior yet, as the policy is expected to be in the initial self-starting phase. Similarly, Stage 1 has a smaller training region (10 x 10 m) compared to other stages, which may have increased the collision rates. Since the agent operates with limited environmental knowledge and exploration space, this instability during the early stage is frequent in reinforcement learning problems. This was later mitigated when the training progressed through increasing environmental complexity and the agent gained knowledge in handling stability, obstacle avoidance, safe navigation, and task-completing behavior.

4.4.1 Overall Curriculum Progression

The curriculum trained policy successfully passed all stage gate criteria at all six stages, with all final gate decisions recorded as ‘advance’ and all stages marked ‘COMPLETE’. The metrics in Table 15 show the successful progression in training with increasing success rates at or above 96% for five of six stages and collision rates at or below 4%. The average episode rewards also increase from 234.25 in Stage 1 to 879.08 in Stage 6. This increase in reward shows that the reward function scales consistently with the environment complexity, providing strong evidence that the policy is improving in navigation.

Also, the gate evaluation metrics align with the statistical analysis of final evaluation metrics across all stages. The selected success episodes are documented in supplementary videos.

Table 15. Curriculum stage performance, including gate-stage metrics.

Metrics	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5	Stage 6
SR	0.81	0.98	0.97	0.99	0.96	0.97
Gate SR	0.87	0.99	0.98	0.98	0.97	0.98
CR	0.09	0.02	0.03	0.01	0.04	0.03
Gate CR	0.08	0.01	0.02	0.02	0.02	0.02

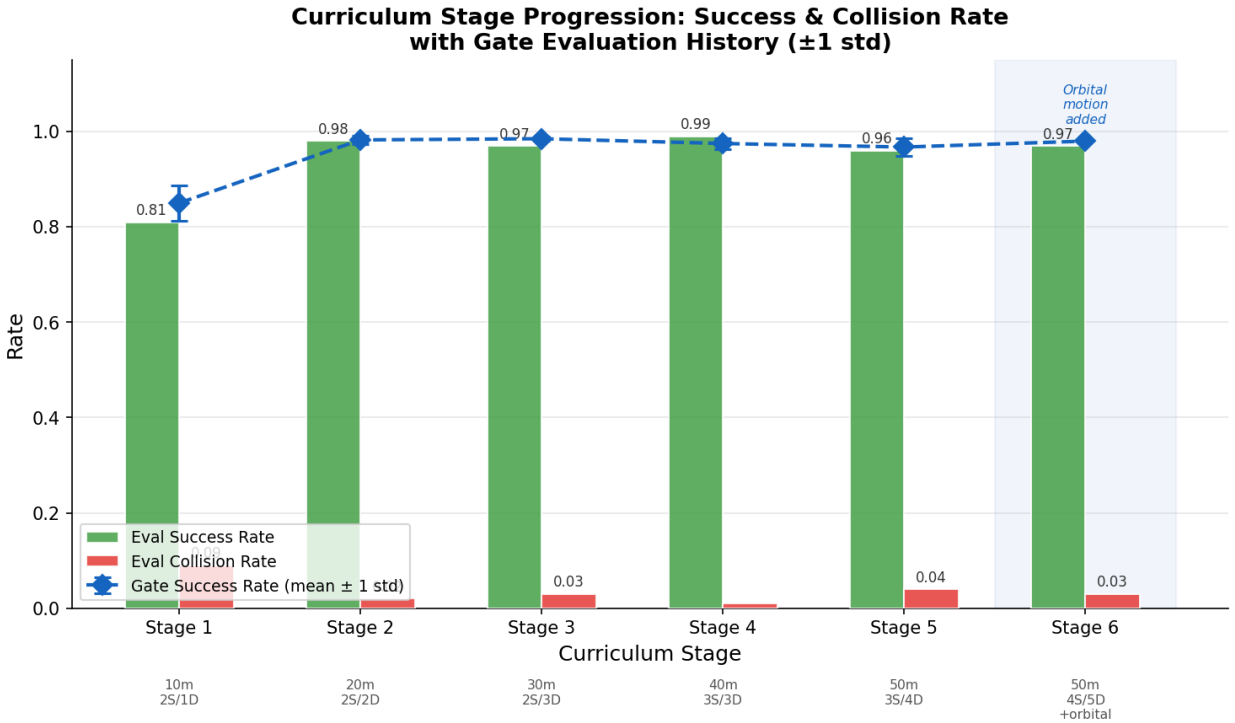


Figure 13. Curriculum stage progression: success rate and collision rate across Stages 1 – 6 with gate success overlaid as a secondary blue line. Error bars show ± standard deviation of gate evaluation history.

4.4.2 Performance Analysis

Regardless of Stage 1 being the simplest configuration in the curriculum, with only 2 static obstacles, 1 dynamic obstacle in a 10 x 10 m arena, and a dynamic speed of only 0.45m/s, it exhibits the lowest success rate (0.81) and highest collision rate (0.09) of all six stages. Similarly, the average episode length of Stage 1 (420.88 steps) is comparable to Stage 5 (416.8 steps), which has more environment complexity in terms of arena size and obstacle count. Trajectory analysis suggested that the smaller arena (10 x 10 m) creates a tightly bounded navigation space. The starting positions x, y ∈ [0.6, 1.5] m place the agent very close to the wall boundaries. The proximity penalty activates as the LiDAR immediately detects wall returns at close range. This creates a consistent negative shaping signal that obstructs goal-relevant learning. Similarly, the episode length irregularity in Stage 1, despite the shorter navigation distance, clearly indicates the inefficient navigation of the agent.

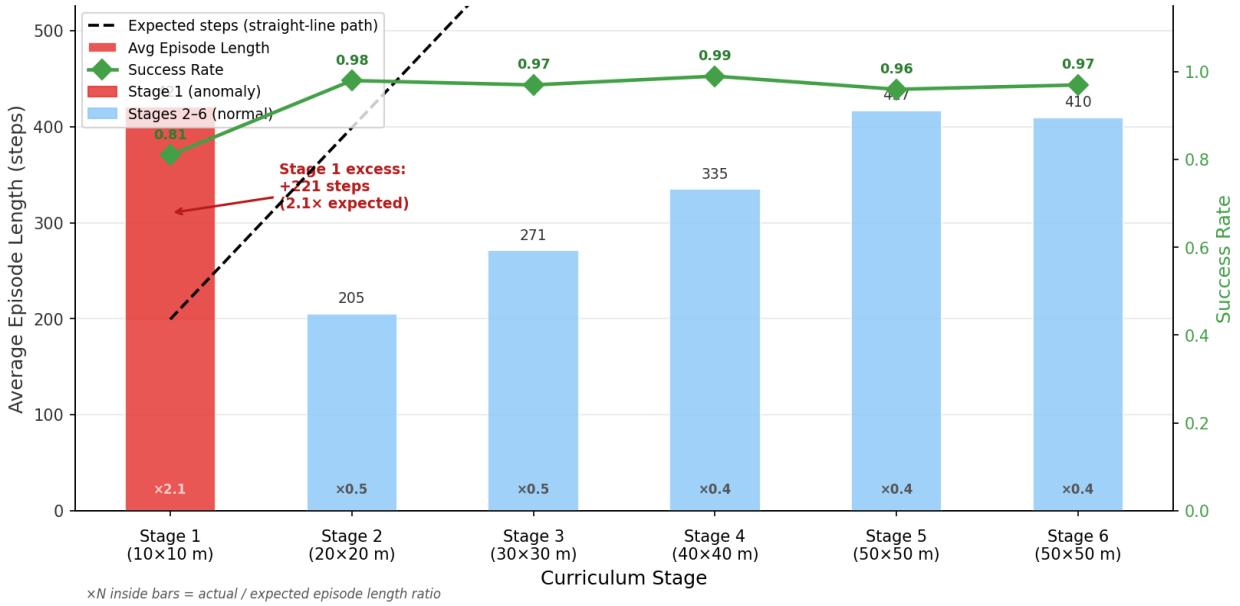


Figure 14. Stage 1 performance analysis: comparison of average episode length across six stages and success rate, showcasing Stage 1's disproportionality with long episodes in a smaller arena. The dashed line shows the expected episode length to the arena size ratio.

Stages 2 through 6 show consistent sequential improvement across all key evaluation metrics. Stage 2 achieves a success rate of 0.98, indicating rapid policy stabilization following Stage 1. This clearly indicates that Stage 1's difficulties are related to arena size rather than policy capability, as the arena expands to 20 x 20 m in Stage 2. Therefore, the agent can learn goal-directed trajectories without consistent proximity penalties from wall returns.

The most demanding training configurations are represented by Stages 4 and 5. Stage 4 achieved the highest success rate (0.99) and collision rate of 0.01 despite the addition of a third static obstacle and dynamic speed of 1.00 m/s. Stage 5 achieved a success rate of 0.96, a collision rate of 0.04, and an average reward of 875.30.

Stage 6 introduced orbital motion at 50% fraction along with a higher obstacle count, including 5 dynamic and 4 static obstacles, a dynamic speed of 1.40 m/s, and achieved a success rate of 0.97 and a collision rate of 0.03. These metrics show an improvement over Stage 5.

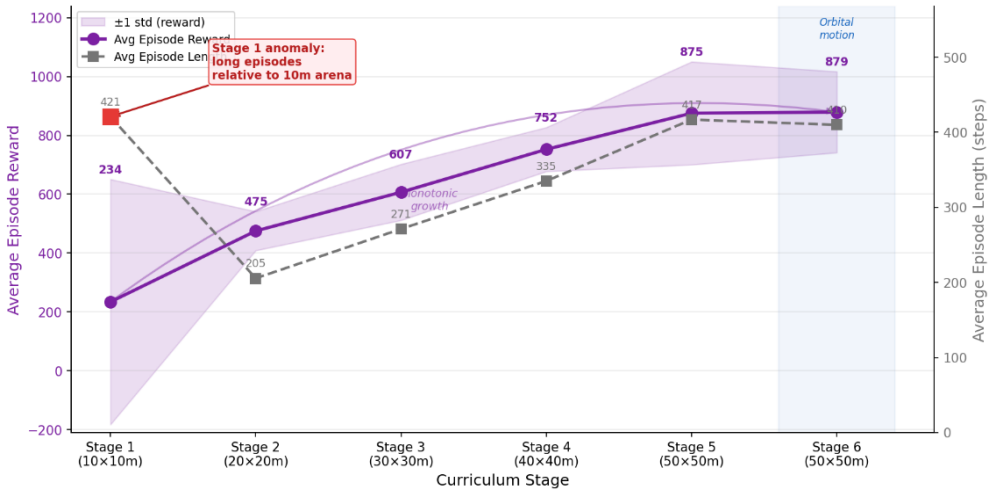


Figure 15. Dual-axis chart: average episode reward and average episode length (grey dashed) across curriculum stages 1- 6, showing sequential reward growth and Stage 1 irregularity.

4.5 Zero-Shot Generalization Results

The evaluation metrics of zero-shot evaluation of the Stage-5 policy across eleven out-of-distribution (OOD) configurations are reported in Table 16. The key metrics are success rate, collision rate, timeout rate, average reward, and average episode length. The evaluation is performed for over 100 episodes per configuration with seed = 99. Stage 5, with a success rate of 0.96 and a collision rate of 0.04, is used as the reference baseline.

Table 16. Zero-shot generalization results

Config	Axis	SR	CI Low	CI High	CR	Timeout rate	Avg reward	Avg length
ZS-D1	Density	0.970	0.936	0.986	0.025	0.005	874.63	418.560
ZS-D2	Density	0.975	0.943	0.989	0.025	0.000	886.55	411.235
ZS-D3	Density	0.955	0.917	0.976	0.045	0.000	870.66	414.470
ZS-S1	Speed	0.990	0.964	0.997	0.010	0.000	897.83	411.450
ZS-S2	Speed	0.990	0.964	0.997	0.010	0.000	899.10	410.705
ZS-A1	Arena	0.520	0.451	0.588	0.480	0.000	175.83	232.700
ZS-A2	Arena	0.870	0.816	0.910	0.045	0.085	1133.06	757.340
ZS-M1	Motion	0.970	0.936	0.986	0.030	0.000	881.30	406.080

ZS-M2	Motion	0.970	0.936	0.986	0.030	0.000	881.42	408.395
ZS-H1	Combined	0.975	0.943	0.989	0.025	0.000	885.17	418.395
ZS-H2	Combined	0.935	0.892	0.962	0.065	0.000	859.73	415.725

4.5.1 Density axis

The three density configurations (ZS-D1, ZS-D2, ZS-D3) achieve success rates of 0.970, 0.975, and 0.955, respectively. All the metrics are above the Stage 5 training baseline, with a success of 0.960. Similarly, ZS-D2 outperforms the baseline by 1.5 percentage points, achieving a success rate of 0.975, with a 95% Wilson CI [0.943, 0.989] indicating a statistically reliable improvement. Whereas ZS-D3 with 5 static and 6 dynamic obstacles achieves a success rate of 0.955, slightly below the baseline but within the Wilson interval margin for 100 episodes.

Overall, these results demonstrate that the LiDAR-based reactive avoidance mechanism improved the obstacle avoidance and helped to generalize well to increased obstacle count. The obstacle count does not qualitatively change the local observation structure as the policy is trained on the 24-ray LiDAR that encodes local proximity.

4.5.2 Speed Axis

Both the speed configurations, ZS-S1 at 1.50 m/s and ZS-S2 at 2.00 m/s, achieved a success rate of 0.990 and a collision rate of 0.010. The success rates matched the best curriculum performance examined in Stage 4 and are the highest among all zero-shot configurations. Similarly, the collision rates of 0.010 correspond to a 75% reduction in relation to the Stage-5 baseline, with a collision rate of 0.040.

From these patterns, it can be derived that the higher obstacle speed does not cause performance degradation. Therefore, it suggests that a policy trained at moderate dynamic speed can generalize to higher speeds (up to 1.67x the training speed in this study) in similar evaluated configurations.

4.5.3 Arena Scale Axis

Most significant performance degradation in zero-shot assessment is revealed in the arena-scale configurations. ZS-A1 with an arena size of 15 x 15 m (smaller arena) achieves a success rate of 0.520 and a collision rate of 0.480. This is the lowest success rate among all zero-shot configurations. The equivalent 95% Wilson interval [0.451, 0.588] is non-intersecting with other configurations' confidence intervals, proving that this result is distinct from others. Similarly, ZS-A2 with an arena size of 70 x 70 m (larger arena) achieves a success rate of 0.870 and a collision rate of 0.045. This configuration has a timeout rate of 0.085, which indicates that the policy struggles to reach the goal within the 1,800-step horizon in a larger arena.

In ZS-A1 (15 x 15 m), the initial position is constrained to a region of only 0.81 m^2 adjacent to the walls according to the methodological setup of x, y ∈ [0.6, 1.5] m. The failure occurs because the policy enters the smaller arena with the safety behaviors calibrated for a larger arena. But in ZS-A2 (70 x 70 m), the issue is different. The goal position is at approximately (56, 56) m and a starting position near (2, 2) m, equivalent to a path length of 76 m, which is longer than the Stage 5 arena of path length ≈65 m. Therefore, the policy avoided the obstacles but failed to complete the tasks within the allowed horizon (1800 steps), resulting in a timeout rate of 0.085.

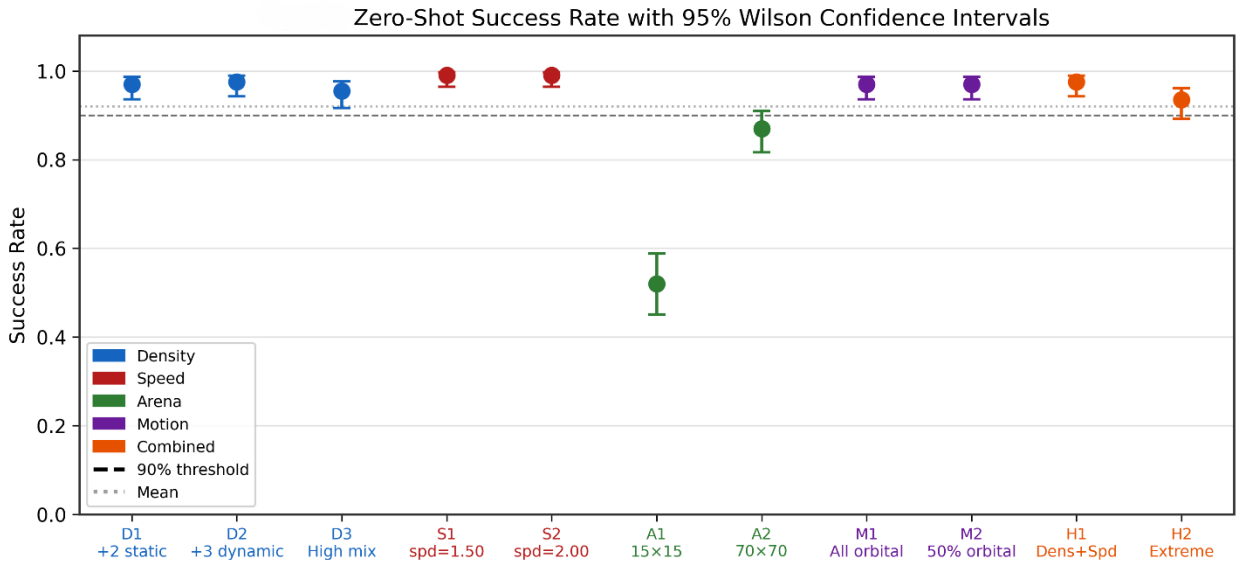


Figure 16. Zero-shot success rate with 95% Wilson CI per configurations

4.5.4 Motion Pattern Axis

Both the motion pattern configurations, ZS-M1 with 100% orbital and ZS-M2 with 50% orbital, achieved a success rate of 0.970 and a collision rate of 0.030. These results mirror the strong performance observed in the density axis exceeding the baseline. Here, the 24-ray LiDAR representation encodes proximity such that the current distances drive the collision avoidance rather than future trajectory predictions. These results are in line with the Stage-6 curriculum results, as the introduction of 50% orbital motion does not cause performance degradation relative to Stage 5.

4.5.5 Combined Hard Axis

The combined configurations tested the interaction of the policy in increased obstacle density and speed. ZS-H1 with 5 static obstacles, 6 dynamic obstacles, and a dynamic speed of 1.60 m/s attained a success rate of 0.975 and a collision rate of 0.025. ZS-H2 with 6 static obstacles, 8 dynamic obstacles, and the highest dynamic speed of 2.00 m/s attained a success rate of 0.935 and a collision rate of 0.065. Also, the 95% interval for ZS-H2 [0.892, 0.962], does not intersect with the upper bounds of speed and motion configurations (≈0.997), showing that the joint effect of extreme density and speed enforces a degradation.

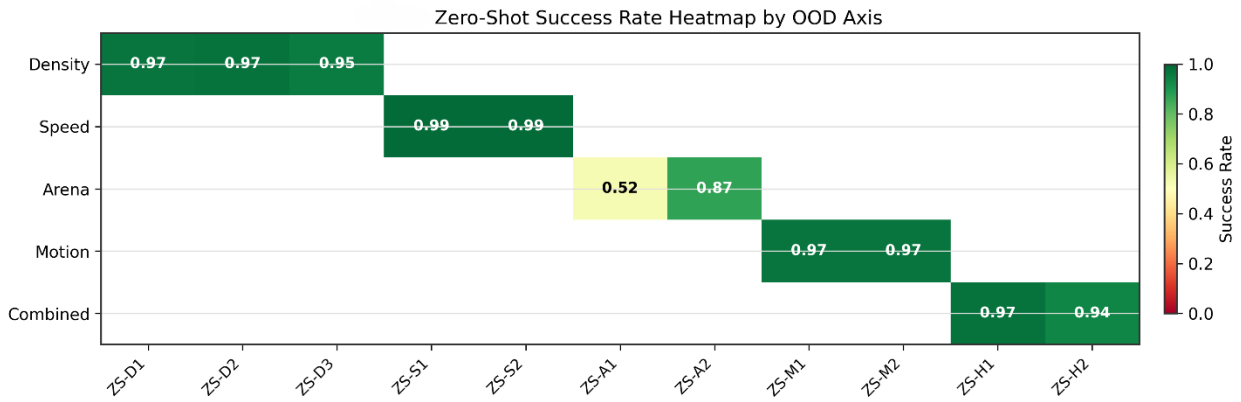


Figure 17. Heatmap of zero-shot success rates organized by axis group and configurations.

4.6 Warm-Up Fine-Tuning Results

For each configuration, zero-shot is used as a baseline (seed = 99), and the post-finetuning uses seed = 2099 across three hard OOD configurations with a 10,000-step warm-up. The warm-up fine-tuning results are shown in Table 17.

Table 17. Warm-up fine-tuning results. Δ SR and Δ CR indicate the change in success and collision rate after fine-tuning.

Config	Phase	SR	CR	Average Reward	Δ SR	Δ CR
WU-D	Zero-shot	0.975	0.025	886.5	-	-
WU-D	After warm-up	0.975	0.025	890.7	+0.000	+0.000
WU-S	Zero-shot	0.990	0.010	899.1	-	-
WU-S	After warm-up	0.950	0.050	859.3	-0.040	+0.040
WU-H	Zero-shot	0.975	0.025	885.2	-	-
WU-H	After warm-up	0.865	0.135	795.9	-0.110	+0.110

4.6.1 WU-D: High Dynamic Density

Fine-tuning the policy on the high dynamic density configuration WU-D with 7 dynamic obstacles for 10,000 steps produced minor changes as the success rate of 0.975 and collision rate of 0.025 remain unchanged. But there was a minor reward improvement from 886.5 to 890.7. These results are reliable with the zero-shot assessment of ZS-D2 that already showed robust performance in this density.

4.6.2 WU-S: High Speed

The high-speed warm-up configurations show an obvious degradation as the success rate decreases from 0.990 to 0.950 and the collision rate increases from 0.010 to 0.050 after fine-tuning for 10,000 steps, resulting in $\Delta SR = -0.040$ and $\Delta CR = +0.040$. These changes show that the previous robust performance is not preserved. Here, the fine-tuning does not provide additional robustness but interrupts an already reliable policy.

4.6.3 WU-H: Combined Hard

The most significant warm-up effect is observed in WU-H, containing 5 static obstacles, 6 dynamic obstacles, and a dynamic speed of 1.60 m/s. After fine-tuning, it shows a drop of success rate from 0.975 to 0.865, and the collision rate increased from 0.025 to 0.135, resulting in ΔSR = −0.110 and ΔCR = +0.110. This represents a significant degradation relative to the zero-shot baseline.

In this setup, the 10,000 gradient steps appeared to be insufficient for a better adaptation to the harder combined distribution. The previous stable avoidance behavior was overwritten, but the updated variables did not converge to a better solution. Overall, these results suggest that a longer warm-up horizon or more advanced fine-tuning methods may be required for hard combined OOD configurations to obtain improvements.

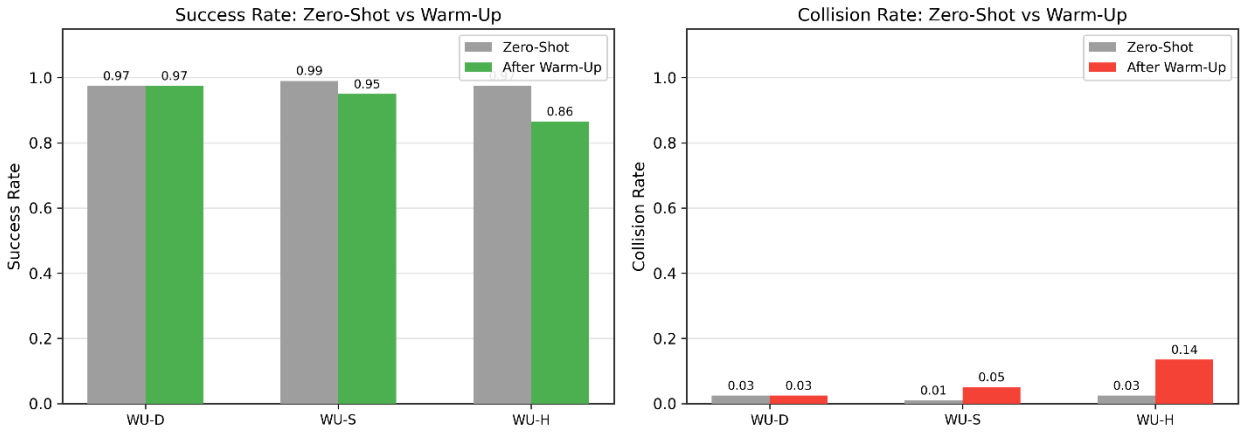


Figure 18. Warm-up before/after comparison: Success rate and collision rate

These results show performance degradation after warm-up fine-tuning in certain environments (WU-S, WU-H). In WU-S, the performance drop occurred mainly due to the overfitting of the policy to the training examples used during fine-tuning, resulting in a reduction in the behavior performance produced by the zero-shot result. The performance drop in WU-H is one of the major drops in the warm-up experiment. Even though the 10,000 gradient step is too short for the adaptation, it is sufficient to disturb the existing behavior that the model had learned. Therefore, resulting in the “catastrophic forgetting” phenomenon as the policy forgets some of the learned behavior upon learning new tasks as the policy enters an unstable middle stage (Kirkpatrick et al., 2017).

4.7 OOD Stress Sweep Results

The stress sweep evaluated the Stage 5 policy across 24 configurations with five difficulty axes, and each varied independently. Table 18. Summarizes the stable region, performance cliff, and breakdown region for each axis.

Table 18. OOD stress sweep summary by axis

Axis	Sweep range	Stable region	Cliff point	Breakdown
Dynamic obstacles	3 – 12	3 – 12 (all)	None	None
Dynamic speed	1.20 – 3.00 m/s	1.20 – 1.80	2.20 m/s	None
Static obstacles	3 – 10	3 – 7	10	None
Arena size	10 – 100 m	50 m only	25 m	10 m, 100m
Orbital fraction	0 – 100%	0 – 25%	50%	75%

4.7.1 Dynamic Obstacle Count

The dynamic obstacle count sweep through ST-DYN-3 to ST-DYN-12 reveals no sudden performance drop. For the counts of 3, 5, 7, 9, and 12, the success rates are 0.980, 1.000, 0.980, 1.000, and 1.000, respectively. The success rate with counts 5, 9, and 12 (1.000) shows the consistency with sampling noise at n=50 rather than systematic reliance on obstacle count. This demonstrated that the policy remains robust even at 12 dynamic obstacles, while the training maximum was 4 (Stage-5).

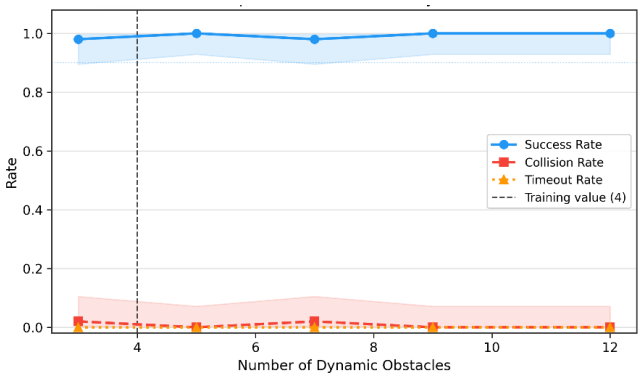


Figure 19. Stress Sweep: Dynamic obstacle count

4.7.2 Dynamic speed

In the speed sweep through ST-SPD-1.20 to ST-SPD-3.00, the success rate remains 0.980 from 1.20 to 1.80 m/s. Then the success rate slightly reduced to 0.960 at 2.20 m/s and returned to .0980 at 3.00 m/s. The Wilson CI [0.865, 0.989] of 2.20 m/s intersected with the confidence intervals at other speeds.

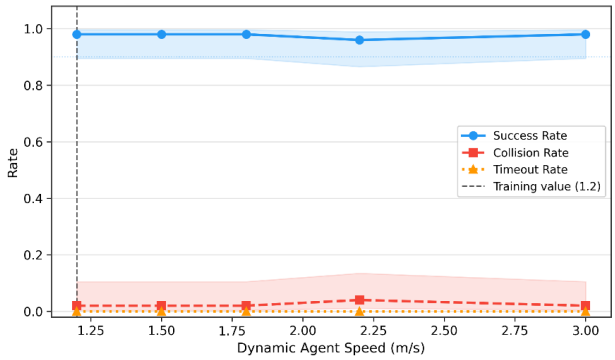


Figure 20. Stress Sweep: Dynamic speed

4.7.3 Static Obstacle Count

The static obstacle count sweep through ST-STA-3 to ST-STA-10 achieved a success rate of 0.980 for 3 – 7 static obstacles and a reduced success rate of 0.960 at 10 static obstacles. The average episode length increased from 409.9 steps at 3 obstacles to 426.4 steps at 10 obstacles. These results indicate that the agent takes a longer path but still completes the task. Also, the collision rate did not exceed 0.040, and no sharp performance threshold was observed.

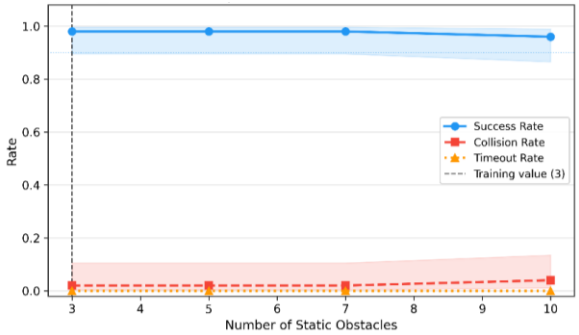


Figure 21. Stress Sweep: Static obstacle count

4.7.4 Arena Size: Bimodal cliff

The arena size sweep reveals the most significant performance cliff in the entire stress sweep assessment. The performance is optimal at training size, ST-SZ-50, with a success rate of 0.980, but degrades in both directions. The success rate drops from 0.980 to 0.180 at 10 x 10 m in ST-SZ-10 with a collision rate of 0.820. At 25 x 25 m in ST-SZ-25, the success rate recovers to 0.700 with a collision rate of 0.280. Then, at a larger arena of 70 x 70 m in ST-SZ-70, attained a success rate of 0.860 with a timeout rate of 0.100, and in 100 x 100 m in ST-SZ-100 attained a success rate of 0.360 with a collision rate of 0.580 and a timeout rate of 0.060.

The smaller arena failures are already identified in ZS-A1, confirming that the agent starts close to the walls, activating strong proximity penalties. But for larger arenas, the issue is different. As the curriculum trained agent was tuned for 50 m arenas, when exposed to larger arenas leads to long trajectories that increase the probability of obstacle collisions, thereby increasing the collision rate rather than timeout rates. Overall, the policy is effective for arena sizes of approximately 40 -70 m, and outside this range causes rapid performance degradation.

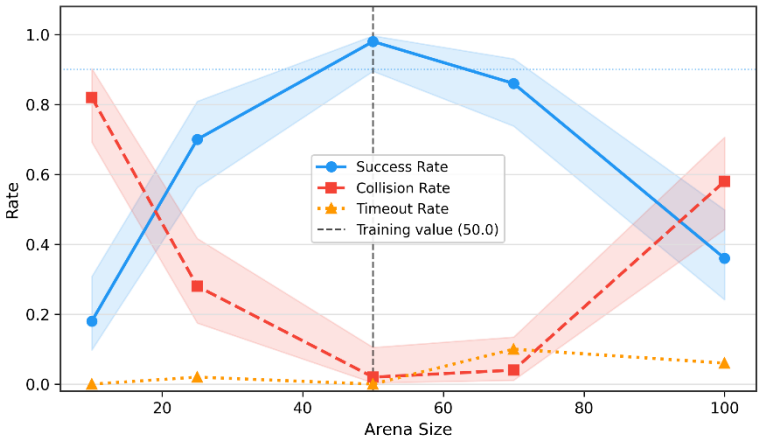


Figure 22. Stress Sweep: Arena size

4.7.5 Orbital Fraction

The orbital fraction sweep reveals a fluctuating pattern. At 0% orbital (training distribution), the success rate is 0.980, improved to a success rate of 1.000 at 25% orbital, dropped to a success rate of 0.940 at 50% and 75% orbital, and returned to 1.000 at 100% orbital. The stabilization at 0.940 for mixed orbital fractions (50%, 75%) suggests that the diverse patterns introduce greater complexity than individual trajectories. A purely orbital environment produces a more spatially regular and predictable obstacle field than a mixed sinusoidal-orbital environment.

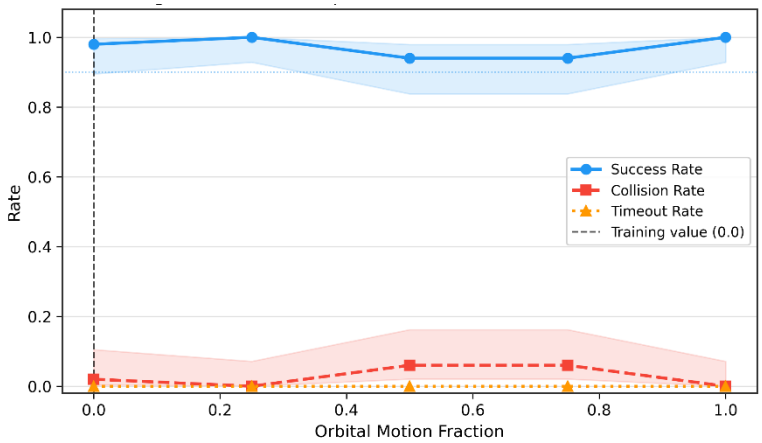


Figure 23. Stress Sweep: Orbital fraction

5 Discussion and Conclusion

5.1 Discussion

This thesis analyzed deep reinforcement learning for mobile robot navigation in dynamic environments, focusing on safety, stability, and generalization. The results clearly demonstrated that an efficient navigation policy depends on multiple factors, including reward design, training procedure, and the variation of evaluation environments, along with the learning algorithm. The experiments conducted revealed that a basic DDPG agent will produce high collision rates, goal-irrelevant navigation, and unstable behavior during evaluation.

The failure behavior analysis of the baseline DDPG policy provided the importance of reward design in efficient navigation. Due to the insufficient reward function, the agent failed to learn goal-oriented motion and followed unsafe trajectories that led to higher collisions with the surrounding wall. The qualitative failure analysis helped to understand why the policy failed, along with providing quantitative metrics showing how often it failed.

In comparison, the training duration is shorter for the initial DDPG baseline experiments. This is a limitation as navigation tasks require longer training episodes. But the purpose of this work was to conduct failure behavior analysis rather than improving DDPG performance while keeping DDPG as a lightweight reference model.

The iterative reward function shaping experiments showed the need for balance while improving the reward design to improve navigation efficiency. The rewards helped in improving motion smoothness, obstacle avoidance, and goal progress, together with safer behavior. On the other hand, if the penalties were too strong, the policy would become less exploratory and overly conservative to avoid collisions by being confined in a position. This reduced the task efficiency and further proved the need to maintain the balance between safety and progression while designing the reward function in navigation.

During the reward refinement experiments, it is noted that the agent learned goal-directed navigation but unsafe termination. It was observed that the agent regularly approached the

goal, but with excessive speed, resulting in collisions after entering the goal radius. This indicates the need to include a quality and safe final approach along with goal completion for a successful navigation. This problem motivated the algorithm enhancement from DDPG to TD3 and SAC for a safer and controlled navigation policy.

A significant liberal modification of reward design across refined version, final safety-aware reward design, algorithm selection, and curriculum training can be observed. It was necessary because the earlier reward structures focused on overcoming limitations like freezing behavior, goal-irrelevant navigation, and the later versions focused on obstacle avoidance, safer navigation, safe termination, and generalization across different complex environments.

Comparison between TD3 and SAC showed that both algorithms are robust candidates for continuous control robot navigation. Even though both algorithms achieved high success rates and low collision rates in controlled static and dynamic environments, SAC provided better performance than TD3 in generalization tests in four unseen environments. This result provides an initial suggestion that SAC may adapt more to unseen environments due to its entropy-based exploration and stochastic policy formation.

The generalization assessment highlights one of the main challenges in deep reinforcement learning for robotics: strong performance in training conditions does not guarantee performance in unseen situations. Even though the policy performed well in trained configurations, the performance degraded as the obstacle density, arena size, or motion dynamics were changed. This confirms the major problems in robot navigation: the sim-to-real transfer and out-of-distribution generalization. In real-world robotic applications, the agent should be robust in both simulation and in varying environmental conditions.

The curriculum learning framework improved the training process by introducing environmental complexities gradually. The agent was able to get introduced to obstacles and learn obstacle avoidance first, then adapt to larger arenas, dynamic obstacles, and varying dynamic speed and motion patterns. The progressive training can improve learning and policy competence. This is confirmed by the increasing trend in average reward along with increasing

success rates across stages. This clearly demonstrated the effectiveness of curriculum learning for complex navigation problems, including layered difficulty and dynamic interaction.

It was observed that Stage 1 attained poor performance among other stages in curriculum learning, even in the simplest environment amid different environmental complexities. This suggested the influence of initial policy development and exploration stability on earlier performance. Effective navigation performance across a complex environment progression will be enabled later once the essential navigation behavior has been learned by the agent.

In parallel, the curriculum results also reveal that robustness is limited to unseen scenarios, i.e., the policy performed well in training stages, but generalization was weaker in unseen distributions. This demonstrated that the curriculum learning is not a complete solution for transferability issues, even though it improves the robustness.

The warm-up fine-tuning was counterproductive at the 10-000 gradient steps, where performance was high at zero-shot, highlighting the need for threshold-triggered fine-tuning strategies. Therefore, while deploying a curriculum-trained policy in an unseen environment, fine-tuning should only be applied after considering the zero-shot performance. If the performance falls below a threshold, together with a larger gradient step budget, it may provide significant results surpassing the zero-shot baseline.

A simplified 2D simulation model, where the robot was described as a spherical agent were used in this study for navigation experiments. This does not fully capture real-world robot characteristics, including kinematic constraints, actuator latency, wheel spin, sensor noise, environmental factors, and irregular localization. Even though the observed curves are useful in understanding reinforcement learning navigation, direct deployment to real-world applications remains limited without extra physical validation and sim-to-real adaptation techniques.

Overall, the results confirm that safe and efficient mobile robot navigation in dynamic environments requires more than a strong reinforcement learning algorithm. For an efficient,

stable policy, it requires a carefully designed reward structure, stable training methods, and evaluation in multiple complex, unseen conditions.

5.2 Conclusion

The scope of this thesis is to analyze and improve deep reinforcement learning-based mobile robot navigation in dynamic environments. Initially started with baseline failure behavior analysis and showed that the standard DDPG without a proper reward structure is inefficient in reliable navigation even in a simple environment, resulting in goal-irrelevant navigation and unstable policy during training and evaluation.

These limitations are addressed by introducing safety-aware reward shaping, later compared SAC and TD3 as strong alternatives. After the algorithm enhancement phase, SAC was selected for curriculum-based training as it achieved better performance than TD3 in the selected four unseen scenario assessments. But, since the number of unseen configurations was limited for the comparison, evaluation across larger and diverse OOD environments is necessary to draw a strong conclusion about the generalization dominance among the two algorithms.

Curriculum-learning experiment demonstrated its effectiveness in helping an agent learn more stable and effective navigation behavior with a gradual increase in the complexity of the environment. The policy attained higher success rates, lower collision rates, and steadily improving reward values across all six stages. This confirmed that training the agent on progressive task complexities prepares the agent for more complex scenarios.

The generalization assessment showed that the trained policy could transfer to unseen environments without retraining, although performance degrades in some difficult scenarios. This means that the proposed method improved the robustness, but full transferability and generalization to unseen environments remain unsolved. This work contributes both a practical training approach and understand the failure modes during generalization assessments.

To summarize, this thesis demonstrated that a combination of reward engineering, stable learning, and curriculum-based training can improve safe navigation in dynamic environments. Due to stronger generalization and stability across the limited number of environments used in this study, SAC proved to be a robust framework in navigation problems.

5.3 Future Work

The future work should extend the current grid-based simulation environment to more continuous, realistic navigation environments. The real-world navigation spaces are rarely regular, even though the current study used structured layouts like 10 x 10, 20 x 20, and larger arena scales. In terms of capturing the obstacle placement variability, motion, and free-space geometry encountered in real-world deployments, continuous environments will outperform regular structured environments.

Controlled ablation studies that isolate the contribution of each reward component should be considered as future work. It should be included to study the impact of individual components on various factors like collision avoidance, smooth navigation, exploration stability, and generalization. This will provide a profound understanding of reward shaping and policy behavior relationship in efficient reinforcement learning navigation.

Another important dimension is improving the generalization. Even though the SAC-based curriculum showed strong performance in training stages and good transferability to unseen environments, the performance still degraded in more difficult out-of-distribution environments. Future studies could therefore incorporate domain randomization, adaptation methods or multi-modal sensing for improving robustness.

Also, the framework could be extended to multi-agent and human-aware socially compliant navigation scenarios, as they introduce additional safety challenges and additional coordination. Furthermore, integrating more robust sensory inputs like depth images or multimodal perception helps to improve obstacle understanding and safer decision-making during navigation.

Even though the simulation results provide a useful foundation, real-world deployment introduces sensor noise, actuation delays, and other practical constraints. Therefore, the future work should consider validation on real robotic hardware to ensure a stronger assessment of the method’s performance in terms of safety and applicability.

References

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning, second edition: An Introduction*. MIT Press.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Graves, A. (2016). Asynchronous methods for deep reinforcement learning. *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, *PMLR 48*, 1928–1937. https://proceedings.mlr.press/v48/mniha16.html

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., & Wierstra, D. (2015). Continuous control with deep reinforcement learning. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.1509.02971

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. https://doi.org/10.48550/arXiv.1707.06347

Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018, July). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning* (pp. 1861-1870). Pmlr.

Fujimoto, S., Hoof, H., & Meger, D. (2018, July). Addressing function approximation error in actor-critic methods. In *International conference on machine learning* (pp. 1587-1596). PMLR.

Zhu, K., & Zhang, T. (2021). Deep reinforcement learning based mobile robot navigation: A review. *Tsinghua Science & Technology*, *26*(5), 674–691. https://doi.org/10.26599/tst.2021.9010012

Zhou, Z., Ren, J., Zeng, Z., Xiao, J., Zhang, X., Guo, X., Zhou, Z., & Lu, H. (2023). A safe reinforcement learning approach for autonomous navigation of mobile robots in dynamic environments. *CAAI Transactions on Intelligence Technology*. https://doi.org/10.1049/cit2.12269

Le, H., Saeedvand, S., & Hsu, C. (2024). A comprehensive review of mobile robot navigation using deep reinforcement learning algorithms in crowded environments. *Journal of Intelligent & Robotic Systems*, *110*(4). https://doi.org/10.1007/s10846-024-02198-w

Everett, M., Chen, Y. F., & How, J. P. (2018). *Motion planning among dynamic, Decision-Making agents with deep reinforcement learning* (pp. 3052–3059). https://doi.org/10.1109/iros.2018.8593871

Everett, M., Chen, Y. F., & How, J. P. (2021). Collision avoidance in Pedestrian-Rich environments with deep reinforcement learning. *IEEE Access*, *9*, 10357–10377. https://doi.org/10.1109/access.2021.3050338

Godoy, J., Karamouzas, I., Guy, S. J., & Gini, M. L. (2016, July). Moving in a Crowd: Safe and Efficient Navigation among Heterogeneous Agents. In *IJCAI* (pp. 294-300). http://dblp.uni-trier.de/db/conf/ijcai/ijcai2016.html#GodoyKGG16

Xiao, X., Liu, B., Warnell, G., & Stone, P. (2022). Motion planning and control for mobile robot navigation using machine learning: a survey. *Autonomous Robots*, *46*(5), 569-597. https://doi.org/10.48550/arxiv.2011.13112

Zhu, Y., Hasan, W. Z. W., Ramli, H. R. H., Norsahperi, N. M. H., Kassim, M. S. M., & Yao, Y. (2025c). Deep Reinforcement Learning of Mobile Robot Navigation in Dynamic Environment: A review. *Sensors*, *25*(11), 3394. https://doi.org/10.3390/s25113394

Levine, S., Finn, C., Darrell, T., & Abbeel, P. (2016). End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, *17*(39), 1-40. https://jmlr.org/papers/v17/15-522.html

Tai, L., Paolo, G., & Liu, M. (2017, September). Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 31-36). IEEE. https://doi.org/10.1109/IROS.2017.8202134

Shakerimov, A., Alizadeh, T., & Varol, H. A. (2023). Efficient Sim-to-Real transfer in reinforcement learning through domain randomization and domain adaptation. *IEEE Access*, *11*, 136809–136824. https://doi.org/10.1109/access.2023.3339568

Gandhi, D., Pinto, L., & Gupta, A. (2017, September). Learning to fly by crashing. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 3948-3955). IEEE. https://doi.org/10.1109/IROS.2017.8206247

Miranda, V. R. F., Neto, A. A., Freitas, G. M., & Mozelli, L. A. (2024). Generalization in deep reinforcement learning for robotic navigation by reward shaping. *IEEE Transactions on Industrial Electronics*, *71*(6), 6013–6020. https://doi.org/10.1109/TIE.2023.3290244

Chahoud, M., Sami, H., Mizouni, R., Bentahar, J., Mourad, A., Otrok, H., & Talhi, C. (2025). Reward shaping in DRL: A novel framework for adaptive resource management in dynamic environments. *Information Sciences*, *715*, 122238. https://doi.org/10.1016/j.ins.2025.122238

Fan, T., Cheng, X., Pan, J., Manocha, D., & Yang, R. (2018, July 19). *CrowdMove: Autonomous mapless navigation in crowded scenarios*. arXiv.org. https://arxiv.org/abs/1807.07870

Rudenko, A., Palmieri, L., Herman, M., Kitani, K. M., Gavrila, D. M., & Arras, K. O. (2020). Human motion trajectory prediction: A survey. *The International Journal of Robotics Research*, *39*(8), 895–935. https://doi.org/10.1177/0278364920917446

Peng, X. B., Andrychowicz, M., Zaremba, W., & Abbeel, P. (2018, May). Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE international conference on robotics and automation (ICRA)* (pp. 3803-3810). IEEE.

Sigal, A., Lin, H. C., & Moon, A. (2023). Improving generalization in reinforcement learning training regimes for social robot navigation. *arXiv preprint* https://arxiv.org/abs/2308.14947

Chukwurah, N., Adebayo, A. S., & Ajayi, O. O. (2024). Sim-to-real transfer in robotics: Addressing the gap between simulation and real-world performance. *International Journal of Robotics and Simulation*, *6*(1), 89-102. https://doi.org/10.1109/ICRA.2018.8460528

Kahn, G., Villaflor, A., Ding, B., Abbeel, P., & Levine, S. (2018, May). Self-supervised deep reinforcement learning with generalized computation graphs for robot navigation. In *2018*

IEEE international conference on robotics and automation (ICRA) (pp. 5129-5136). IEEE. https://doi.org/10.1109/ICRA.2018.8460655

Macenski, S., Foote, T., Gerkey, B., Lalancette, C., & Woodall, W. (2022). Robot Operating System 2: Design, architecture, and uses in the wild. *Science Robotics*, *7*(66), eabm6074. https://doi.org/10.1126/scirobotics.abm6074

Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., & Riedmiller, M. (2014, January). Deterministic policy gradient algorithms. In *International conference on machine learning* (pp. 387-395). Pmlr. https://proceedings.mlr.press/v32/silver14.html

Schulman, J., Levine, S., Abbeel, P., Jordan, M., & Moritz, P. (2015, June). Trust region policy optimization. In *International conference on machine learning* (pp. 1889-1897). PMLR. https://proceedings.mlr.press/v37/schulman15.html

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533. https://doi.org/10.1038/nature14236

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D., & Hadsell, R. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, *114*(13), 3521–3526. https://doi.org/10.1073/pnas.1611835114

Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M., & Freitas, N. (2016, June). Dueling network architectures for deep reinforcement learning. In *International conference on machine learning* (pp. 1995-2003). PMLR. https://proceedings.mlr.press/v48/wangf16.html

Bellemare, M. G., Dabney, W., & Munos, R. (2017, July). A distributional perspective on reinforcement learning. In *International conference on machine learning* (pp. 449-458). Pmlr. https://proceedings.mlr.press/v70/bellemare17a.html

Hessel, M., Modayil, J., Van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., ... & Silver, D. (2018, April). Rainbow: Combining improvements in deep reinforcement learning.

In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 32, No. 1). https://doi.org/10.1609/aaai.v32i1.11796

Dabney, W., Rowland, M., Bellemare, M., & Munos, R. (2018, April). Distributional reinforcement learning with quantile regression. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 32, No. 1). https://doi.org/10.1609/aaai.v32i1.11791

Sun, C., Wang, R., Li, Q., & Hu, X. (2021). Reward space noise for exploration in deep reinforcement learning. *International Journal of Pattern Recognition and Artificial Intelligence*, *35*(10), 2152013. https://doi.org/10.1142/S0218001421520133

Osband, I., Blundell, C., Pritzel, A., & Van Roy, B. (2016). Deep exploration via bootstrapped DQN. *Advances in neural information processing systems*, *29*.

Garcıa, J., & Fernández, F. (2015). A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, *16*(1), 1437-1480.

Achiam, J., Held, D., Tamar, A., & Abbeel, P. (2017, July 17). *Constrained policy optimization*. PMLR. https://proceedings.mlr.press/v70/achiam17a.html

Tessler, C., Mankowitz, D. J., & Mannor, S. (2018). Reward constrained policy optimization. *arXiv preprint arXiv:1805.11074*.

Brunke, L., Greeff, M., Hall, A. W., Yuan, Z., Zhou, S., Panerati, J., & Schoellig, A. P. (2022). Safe learning in Robotics: From Learning-Based Control to Safe Reinforcement Learning. *Annual Review of Control Robotics and Autonomous Systems*, *5*(1), 411–444. https://doi.org/10.1146/annurev-control-042920-020211

Berkenkamp, F., Turchetta, M., Schoellig, A., & Krause, A. (2017). Safe model-based reinforcement learning with stability guarantees. *Advances in neural information processing systems*, *30*.

Chow, Y., Tamar, A., Mannor, S., & Pavone, M. (2015). Risk-sensitive and robust decision-making: a cvar optimization approach. *Advances in neural information processing systems*, *28*.

Chen, C., Liu, Y., Kreiss, S., & Alahi, A. (2019). Crowd-Robot Interaction: Crowd-Aware Robot Navigation With Attention-Based Deep Reinforcement Learning. *IEEE Robotics and Automation Letters*, 6015–6022. https://doi.org/10.1109/icra.2019.8794134

Long, P., Fanl, T., Liao, X., Liu, W., Zhang, H., & Pan, J. (2018). Towards Optimally Decentralized Multi-Robot Collision Avoidance via Deep Reinforcement Learning. *ICRA*, 6252–6259. https://doi.org/10.1109/icra.2018.8461113

Sartoretti, G., Kerr, J., Shi, Y., Wagner, G., Kumar, T. K. S., Koenig, S., & Choset, H. (2019). PRIMAL: pathfinding via reinforcement and imitation Multi-Agent learning. *IEEE Robotics and Automation Letters*, *4*(3), 2378–2385. https://doi.org/10.1109/lra.2019.2903261

Gupta, S., Davidson, J., Levine, S., Sukthankar, R., & Malik, J. (2017). Cognitive mapping and planning for visual navigation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2616-2625).

Savinov, N., Dosovitskiy, A., & Koltun, V. (2018). Semi-parametric topological memory for navigation. *arXiv preprint arXiv:1803.00653*.

Kunhoth, J., Karkar, A., Al-Maadeed, S., & Al-Ali, A. (2020). Indoor positioning and wayfinding systems: a survey. *Human-centric Computing and Information Sciences*, *10*(1), 18. https://doi.org/10.1186/s13673-020-00222-0

Chaplot, D. S., Gandhi, D. P., Gupta, A., & Salakhutdinov, R. R. (2020). Object goal navigation using goal-oriented semantic exploration. *Advances in Neural Information Processing Systems*, *33*, 4247-4258. https://proceedings.neurips.cc/paper/2020/hash/2c75cf2681788adaca63aa95ae028b22-Abstract.html

Kong, H., Xing, Q., Wang, Q., Niu, R., Chen, H., Wang, Y., Wang, S., Duan, Z., & Chang, Y. (2025). ADAC: Actor-Double-Attention-Critic for Multi-Agent Cooperation in Mixed Cooperative-Competitive environments. *IEEE Transactions on Intelligent Transportation Systems*, *26*(7), 9579–9592. https://doi.org/10.1109/tits.2025.3562302

Foerster, J., Assael, I. A., De Freitas, N., & Whiteson, S. (2016). Learning to communicate with deep multi-agent reinforcement learning. *Advances in neural information processing systems*. https://proceedings.neurips.cc/paper/2016/hash/c7635bfd99248a2cdef8249ef7bfbef4-Abstract.html

Rashid, T., Samvelyan, M., Schroeder de Witt, C., Farquhar, G., Foerster, J., & Whiteson, S. (2018). QMIX: Monotonic value function factorisation for deep multi-agent reinforcement

learning. *Proceedings of the 35th International Conference on Machine Learning (ICML)*, *PMLR 80*, 4295–4304. https://proceedings.mlr.press/v80/rashid18a.html

Iqbal, S., & Sha, F. (2019). Actor-attention-critic for multi-agent reinforcement learning. *Proceedings of the 36th International Conference on Machine Learning (ICML)*, *PMLR 97*, 2961–2970. https://proceedings.mlr.press/v97/iqbal19a.html

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., & Abbeel, P. (2017). Domain randomization for transferring deep neural networks from simulation to the real world. *IROS*. https://doi.org/10.1109/IROS.2017.8202133

James, S., Davison, A. J., and Johns, E. (2019). Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 12627–12637.

Liu, Q., Sun, Z., Cui, Y., Li, H., Li, G., Shao, L., Chen, J., & Ye, Q. (2026a). DeXREPNet++: Learning dexterous robotic manipulation with geometric and spatial Hand-Object representations. *IEEE Transactions on Robotics*, *42*, 799–818. https://doi.org/10.1109/tro.2026.3651669

OpenAI, Akkaya, I., Andrychowicz, M., Chociej, M., Litwin, M., McGrew, B., Petron, A., Paino, A., Plappert, M., Powell, G., Ribas, R., Schneider, J., Tezak, N., Tworek, J., Welinder, P., Weng, L., Yuan, Q., Zaremba, W., & Zhang, L. (2019). Solving Rubik's Cube with a Robot Hand. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.1910.07113

Rusu, A. A., Večerík, M., Rothörl, T., Heess, N., Pascanu, R., & Hadsell, R. (2017, October 18). *Sim-to-Real Robot Learning from Pixels with Progressive Nets*. PMLR. https://proceedings.mlr.press/v78/rusu17a.html

Tan, J., Zhang, T., Coumans, E., Iscen, A., Bai, Y., Hafner, D., Bohez, S., & Vanhoucke, V. (2018). Sim-to-Real: Learning Agile Locomotion for Quadruped Robots. *Proceedings of Robotics: Science and Systems (RSS)*. https://doi.org/10.15607/RSS.2018.XIV.010

Hwangbo, J., Lee, J., Dosovitskiy, A., Bellicoso, D., Tsounis, V., Koltun, V., & Hutter, M. (2019). Learning agile and dynamic motor skills for legged robots. *Science Robotics*, *4*(26), eaau5872. https://doi.org/10.1126/scirobotics.aau5872

Bengio, Y., Louradour, J., Collobert, R., & Weston, J. (2009, June). Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning* (pp. 41-48). https://doi.org/10.1145/1553374.1553380

Florensa, C., Held, D., Geng, X., & Abbeel, P. (2018, July). Automatic goal generation for reinforcement learning agents. In *International conference on machine learning* (pp. 1515-1528). PMLR. https://proceedings.mlr.press/v80/florensa18a.html

Pathak, D., Agrawal, P., Efros, A. A., & Darrell, T. (2017, July). Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning* (pp. 2778-2787). PMLR. https://proceedings.mlr.press/v70/pathak17a.html

Burda, Y., Edwards, H., Pathak, D., Storkey, A., Darrell, T., & Efros, A. A. (2018). Large-scale study of curiosity-driven learning. *arXiv preprint arXiv:1808.04355*.

Ecoffet, A., Huizinga, J., Lehman, J., Stanley, K. O., & Clune, J. (2021). First return, then explore. *Nature*, *590*(7847), 580-586.

Schmidhuber, J. (2010). Formal theory of creativity, fun, and intrinsic motivation (1990–2010). *IEEE transactions on autonomous mental development*, *2*(3), 230-247. https://doi.org/10.1109/TAMD.2010.2056368

Oudeyer, P. Y., & Kaplan, F. (2007). What is intrinsic motivation? A typology of computational approaches. *Frontiers in neurorobotics*, *1*, 108.

Forestier, S., Portelas, R., Mollard, Y., & Oudeyer, P. Y. (2022). Intrinsically motivated goal exploration processes with automatic curriculum learning. *Journal of Machine Learning Research*, *23*(152), 1-41.

Appendices

Appendix A: Code Repository and Supplementary Videos

This appendix provides access to the full implementation code and supplementary video material that show the experimental results reported in this thesis. The repository contains environment configurations, training scripts, evaluation scripts, and visualizations.

The supplementary videos include the failure behaviour episode, the combined curriculum stage results, and episodes for three generalization assessment modes.

A.1 GitHub Repository

Repository: https://github.com/SonasWilson/thesis-robot-navigation

Repository contents:

- Environment configuration and setup file
- Training code for DDPG, TD3, and SAC
- Curriculum training scripts
- Generalization evaluation scripts
- Plotting scripts

A.2 Supplementary videos

1. Failure behaviour (wall collision) video: https://youtu.be/stY_dH3zB6c
2. Curriculum learning success navigation: https://youtu.be/96WW10VOvDY
3. Generalization Test (Zero-Shot): https://youtube.com/shorts/5H5sKyhUbdk?feature=share
4. Generalization Test (Warm-Up Fine-Tuning): https://youtube.com/shorts/_pe7-k8CHos?feature=share
5. Generalization Test (OOD Stress Sweep): https://youtu.be/fk9_Z5pCRbY