Honors Thesis Proposal

For

SwarmBench: A Benchmarking Framework for Multi-Agent Reinforcement Learning in Multi-Robot Systems

Akshat Guduru

Chinwendu Enyioha

Chinwendu Enyioha, Ph.D.
Thesis Committee Chair
Department of Electrical and Computer Engineering

Manoj Chopra

Manoj Chopra, Ph.D.
HUT Liaison
Department of Electrical and Computer Engineering

yunjunxu

Yunjun Xu, Ph.D.
Thesis Committee Member
Department of Aerospace and Mechanical Engineering

SwarmBench: A Benchmarking Framework for Multi-Agent Reinforcement Learning in Multi-Robot Systems

Honors Undergraduate Thesis Proposal

Akshat Guduru

Department of Computer Science

University of Central Florida

November 2025

1 Abstract

Multi-agent reinforcement learning (MARL) has gained attention as a promising approach for coordinating teams of robots in cooperative and competitive scenarios such as warehouse navigation, traffic management, and search-and-rescue. However, there remains a significant gap in the evaluation and benchmarking of MARL algorithms: existing frameworks primarily rely on toy simulations or game-based domains, while robotics labs often maintain custom ROS-based setups that make results difficult to reproduce and compare. Furthermore, there lacks a unified framework that allows for online and offline training regimes, communication instability, and realistic continuous-control settings. This thesis proposes to design and implement a modular benchmarking framework for MARL in multi-robot environments. The framework will define a standard task specification and API, implement a set of multi-robot navigation and coordination tasks, and provide modules for generating offline datasets, injecting communication constraints, and varying observation impairments. Using this framework, we will benchmark several representative MARL algorithms and analyze their performance, safety, and robustness under different data and communication constraints. The overall framework attempts to improve the reproducibility of MARL research for robotic swarms and offer guidance on which methods are the most reliable under realistic multi-robot constraints.

2 Introduction

2.1 Background and Motivation

Coordinating many robots to work together in complex, uncertain environments is a central challenge in modern robotics. Examples include controlling fleets of warehouse robots routing items through narrow aisles, autonomous vehicles interacting in traffic, and aerial or ground robots collaborating in search-and-rescue missions. In all of these settings, each robot must make decisions based on partial, noisy information while also avoiding collisions, respecting physical constraints, and cooperating with other robots to achieve shared goals. Designing control policies by hand for such tasks quickly becomes infeasible as the number of robots and the complexity of the environment grows.

Multi-Agent Reinforcement Learning (MARL) has emerged as a promising solution for addressing these challenges. In MARL, multiple decision-making agents learn policies through interaction with an environment, receiving feedback in the form of rewards or penalties. When applied to multi-robot systems, MARL offers potential to automatically discover decentralized control strategies that are adaptive, robust, and capable of exploiting coordination and communication between robots. Recent research has demonstrated that MARL can be effective for tasks such as cooperative navigation, formation control and coverage [1], suggesting that it may play an important role in future multi-robot systems.

2.2 MARL for Multi-Robot Systems

At a high level, MARL extends the standard reinforcement learning framework to settings with multiple agents that act and learn simultaneously through interactions with the environment. These environments are often modeled as a Markov Game, a multi-agent extension of traditional Markov Decision Processes, where the joint states, joint actions, and joint reward dynamics depend on the behavior of all agents. This introduces several difficulties that do not arise in the single-agent case, including non-stationarity (because each agent's learning changes the effective environment for the others), partial observability, and credit assignment across agents.

To manage these challenges, a variety of MARL approaches have been developed. Centralized Training Decentralized Execution (CTDE) methods learn joint value functions or policies using global information during training but deploy decentralized policies at test time. Value-decomposition methods, such as Value Decomposition Networks (VDN) and QMIX, approximate the joint value function as a combination of per-agent value functions, enabling coordinated behavior while preserving decentralized execution [2, 3]. Policy-gradient methods, such as Multi-Agent Proximal Policy Optimization (MAPPO) and variants, directly opti-

mize parameterized policies for each agent [4]. Other lines of work explicitly model communication between agents or incorporate safety constraint handling into the learning process. Many of these methods have been applied, at least in simulation, to multi-robot scenarios where agents must navigate, avoid collisions, and achieve shared objectives.

2.3 Current Benchmarking Practices and Limitations

Despite this progress, the way MARL algorithms are evaluated and benchmarked remains fragmented. Much of the empirical work in MARL relies on game-like or toy domains, such as grid-worlds, simple particle environments, or competitive games that are far removed from the dynamics and constraints of real-world robots. General-purpose MARL benchmark frameworks provide useful standardization for algorithm comparison, but they rarely account for realistic robot kinematics, sensing, or communication constraints. As a result, it is difficult to know how well algorithms that perform well on these benchmarks will transfer to multi-robot control problems.

In parallel, the robotics community has developed a variety of RL-robotics frameworks and ROS-based tools that make it possible to train and develop learning-based controllers on real or simulated robots. However, these setups are often designed for single-robot tasks or for specific platforms or laboratories. Multi-robot benchmarking tends to rely on custom ROS and simulator configurations, with task definitions, sensor models, and evaluation metrics varying widely between projects. This heterogeneity makes it challenging to reproduce results, compare algorithms across labs, or systematically study how MARL methods behave under realistic constraints such as unreliable communication, limited data, or complex continuous-control dynamics.

A further limitation is that most existing evaluation frameworks treat online and offline learning, communication reliability, and observation fidelity as separate concerns. For example, offline RL benchmarks typically focus on single-agent domains with fixed datasets, while communication-robust MARL studies introduce ad hoc message-drop models specific to their own environments. High-dimensional sensing, such as LiDAR or camera-based observations, is often explored in isolated robotics studies rather than as part of a unified MARL benchmark suite.

2.4 Problem Statement

Taken together, these issues point to a clear gap. There is currently no unified, open benchmarking framework that specifically targets multi-robot MARL tasks and supports, within a single set of task definitions, (i) both online and offline training regimes, (ii) systematic variation of communication reliability and bandwidth, and

(iii) realistic continuous-control settings with configurable observation models. Instead, MARL algorithms are typically evaluated either on abstract, game-like environments or on bespoke multi-robot setups that are difficult to replicate and can be expensive if damaged during experiments. This greatly hinders the ability of researchers to compare methods fairly, to understand the conditions under which particular algorithms are robust or fragile, and to build on each other's work when moving toward real-world robot swarms.

2.5 Proposed Contributions and Structure of the Proposal

The main expected contributions of this thesis are:

- A standardized task specification and API for cooperative multi-robot MARL tasks, designed to support multiple simulation backends and observation modalities.
- A set of benchmark multi-robot navigation and coordination tasks implemented within this framework, suitable for online MARL training and evaluation.
- Modules for generating, storing, and reusing offline multi-agent datasets, enabling offline MARL experiments and systematic comparison of algorithms in the offline setting.
- A communication robustness layer that allows controlled variation of message reliability, bandwidth, and network topology as part of the benchmark configuration.
- An empirical study comparing several representative MARL algorithms along axes of performance, safety, and robustness under different data and communication regimes.

The remainder of this proposal is organized as follows. Section 3 reviews background and related work on MARL, multi-robot control, and existing benchmarks and frameworks. Section 4 then describes the proposed framework design, tasks, algorithms, and experimental methodology in detail.

3 Literature Review

3.1 Multi-Agent Reinforcement Learning Algorithms

Most of the algorithms that will be evaluated in this thesis follow the CTDE paradigm. Under CTDE, agents are trained using additional information such as the global state or other agents' actions, but are constrained to use only local observations at execution time. This paradigm is particularly relevant to multi-robot systems, where richer state information may only be available in simulation or during training, while deployed robots must act based on their own onboard sensors and limited communication.

Lowe et al. introduced Multi-Agent Deep Deterministic Policy Gradients (MADDPG) as a general purpose actor-critic framework for mixed cooperative-competitive environments [1]. MADDPG assigns each agent its own deterministic policy (actor) but uses a centralized critic that conditions on the joint observations and joint actions, mitigating non-stationary caused by simultaneous learning. The paper demonstrates that this approach can solve a variety of particle-world tasks, including cooperative navigation and competitive scenarios, and the associated Multi-Agent Particle Environment (MPE) has since become a standard MARL benchmark. MADDPG is an important reference point because it exemplifies the CTDE idea and provides early evidence that centralized critics can stabilize multi-agent learning.

For fully cooperative settings with a single team reward, value-decomposition methods aim to learn a joint action-value function that can be factored into per-agent utilities. Sunehag et al. proposed Value-Decomposition Networks (VDN), which approximate the joint Q-function as a simple sum of individual agent value functions [2]. This factorization allows each agent to condition only on its local observation while still optimizing a global team objective, and helps alleviate issues such as the "lazy agent" problem where some agents exploit others' contribution to receive high reward without acting. VDN provides a simple but effective baseline for cooperative MARL and motivates more expressive factorization schemes.

Building on this idea, Rashid et al. introduced QMIX, a monotonic value factorization method that uses a mixing network to combine per-agent value functions into a joint action-value [3]. The mixing network is constrained to be monotonic in each agent's utility, which ensures that selecting each agent's greedy action with respect to its individual value function is consistent with maximizing the joint Q-value. This constraint preserves decentralization while allowing more flexible interactions between agents than a simple sum. QMIX achieved state-of-the-art results on the StarCraft Multi-Agent Challenge (SMAC) making it a widely used baseline for cooperative CTDE methods and an important candidate for evaluation in any MARL benchmark.

In contrast to value-based methods, policy-gradient approaches directly optimize parameterized policies, often with a shared centralized value function. Yu et al. showed that a carefully implemented multi-agent variant of Proximal Policy Optimization (MAPPO) can be surprisingly competitive across multiple cooperative MARL benchmarks, including MPE, SMAC, and Hanabi [4]. MAPPO adopts a centralized critic that conditions on the global state while maintaining decentralized policies at execution, and achieves strong performance and reasonable sample efficiency without extensive domain-specific tuning. Given its simplicity and strong emprical results, MAPPO has emerged as a de facto on-policy baseline for cooperative MARL.

Together, these algorithms, represent the dominant families of deep MARL methods used in contemporary benchmarks: actor-critic CTDE methods, additive value-decomposition, monotonic mixing-based value-decomposition, and policy-gradient approached with centralized critics. SwarmBench will focus on evaluating

representative algorithms from these families on multi-robot tasks, with particular attention to how their performance and robustness change under different data regimes, communication conditions, and observation models.

3.2 Existing MARL Benchmarks and API

A major driver of recent progress in deep MARL has been the availability of standardized benchmark environments and APIs. PettingZoo provides a widely used interface for multi-agent environments, analogous to the role played by OpenAI Gym in single-agent reinforcement learning [5]. It groups environments into collection such as classical games, Atari, the Multi-Agent Particle Environments (MPE), and others, and exposes a consistent agent-iteration API that simplifies training and evaluation across domains. By decoupling algorithm implementations from environment specifics, PettingZoo has enables more systematic comparison of MARL methods and has become a de facto standard for experimental work in the field.

The OpenAI MPE paper originally introduced alongside MADDPG, is one of the most influential environment suites in MARL [1]. MPE consists of a set of simple, continuous-state-particle-world scenarios, including cooperative navigation, predator-prey, and communication=based tasks. Each agent is represented as a point mass moving in a two-dimensional space, with simple physics and handcrafted reward structure. Despite their simplicity, these tasks expose important multi-agent challenges, such as partial observability, and they have been adopted in numerous subsequent studies as a standard testbed for cooperative and mixed-motive MARL.

Beyond MPE, other benchmark suites have targeted specific multi-agent settings. The StarCraft Multi-Agent Challenge (SMAC) focuses on micromanagement in real-time strategy combat, where each agent controls an individual unit in a StarCraft II scenario [6]. These benchmarks broaden the range od MARL evaluation tasks, but they remain largely game or simulation centric and abstract away many aspects of real robot dynamics and sensing.

Taken together, PettingZoo, MPE, SMAC, and related suites have established a culture of standardized reusable environments that greatly benefits MARL research. However, they primarily focus on non-robotic domains, with simplified dynamics, idealized communication and observation models. They offer limited support for realistic continuous-control tasks grounded in multi-robot systems, and they do not explicitly address axes that are crucial for deployment, such as varying communication reliability, offline training from logged trajectories, or integration with robotics middleware like ROS. These limitations is what motivated our development of a complementary benchmark framework that is tailored to multi-robot MARL while retaining the benefits of a standardized API.

3.3 Multi-Robot RL and Sim-To-Real Frameworks

While general-purpose MARL benchmarks focus largely simpler game-based environments, a parallel line or work has been developed frameworks that more directly target multi-robot systems. These frameworks aim to bridge the gap between learning algorithms and realistic robot dynamics, sensing, and deployment, often providing both simulation environment and pathways to real hardware.

Vectorized Multi-Agent Simulator (VMAS) is an open-source framework designed for efficient MARL benchmarking in multi-robot scenarios [7]. VMAS provides a 2D physic engine implemented in PyTorch and a set of twelve multi-robot environments that test challenges such as coordination, communication, and adversarial interactions. Its key contribution is vectorization: thousands of environment instances can be simulated parallel on accelerators, yielding over 100× speedups compared to MPE for large batches [7]. VMAS thus addresses the need for scalable training in collective robot-learning tasks, but it focuses on a single, 2D simulated physics backend and does not directly integrate with robotics middleware such as ROS or real robot platforms.

MultiRobotLearn take a different approach by explicitly unifying simulation and physical multi-robot deployments [8]. It provides a framework for deep reinforcement learning with multiple robots that supports both discrete and continuous action spaces, with standardized simulated scenarios that can be deployed to real-world environments with minimal changes. The authors demonstrate that the framework on two real-world scenarios with minimal changes, showing how policies trained in simulation can transfer to hardware when appropriate abstractions are used [8]. MultiRobotLearn thus addresses the sim-to-real gap for multi-robot systems, but it is primarily a development framework rather than a widely adopted benchmark suite with standardized tasks and metrics across multiple axes of evaluation.

SMART (Scalable Multi-Robot Reinforcement Learning Platform) further emphasizes the evaluation side of multi-robot RL [9]. It consists of a simulation environment with complex integration scenarios and a real-world multi-robot testbed, together forming an emulation platform for multi-robot reinforcement learning. SMART exposes plug-and-play agent-environment APIs and provides benchmarking tasks, such as cooperative driving lane change, for comparing MRRL (Multi-Robot Reinforcement Learning) algorithms [9]. However, SMART is tailored to specific domains and hardware configurations, and it does not attempt to define a general multi-robot MARL benchmark that is easily portable across labs.

Gym-Ignition focuses on reproducible robotic simulations for reinforcement learning by interfacing with the Ignition Gazebo simulation [10]. It provides abstractions for robots and tasks, exposing an OpenAI Gym-compatible interface that supports multiple physics engine, accelerated simulation, and distributed execution [10]. Gym-Ignition is especially notable for its emphasis on modularity and reproducibility, as well

as its ability to run the same task definitions in simulation and on real robots when appropriate runtime backends are provided. Nonetheless, it is largely single-agent in emphasis and does not target multi-agent nor multi-robot benchmarking as a primary objective.

In summary, VMAS, MultiRobotLearn, SMART, and Gym-Ignition collectively demonstrate that there is strong interest in frameworks that connect deep RL with realistic multi-robot systems. VMAS provides scalable, vectorized multi-robot simulations; MultiRobotLearn and SMART bridge simulation with real robot platforms; and Gym-Ignition offers reproducible, modular robotic environments. However, none of these frameworks are designed as a unified. open benchmarking suite specifically for multi-robot MARL that systematically varies online versus offline training, communication reliability, and observation fidelity under a common task specification and API. This gap motivates the design of SawrmBench as a Complementary benchmark focused on multi-robot MARL, building on the lessons of these frameworks while targeting standardized evaluation along axes thar are critical for deployment.

3.4 Offline RL and Offline MARL Benchmarks

In addition to online learning in simulators or on physical robots, there has been growing interest in *offline* reinforcement learning, where policies are trained purely from fixed datasets of past experience. Offline RL is appealing for robotics and multi-robots systems because it reduces the need for risky or expensive online exploration and enables reuse of previously collected data. However, offline learning is also more sensitive to dataset coverage and distributional shift, which makes careful benchmarking essential.

D4RL (Datasets for Deep Data-Driven Reinforcement Learning) established one of the first comprehensive benchmark suites for offline RL [11]. It provides standardized datasets and evaluation protocols for a range of domains, including locomotion, manipulation, and navigation, with data collected from a mixture of random, suboptimal, and expert policies. D4RL has become a default testbed for single-agent settings and dos not address multi-agent coordination, communication, or the specific challenges of multi-robot systems.

More recently, the OG-MARL benchmark (Off-the-Grid MARL) extended the offline RL paradigm to cooperative multi-agent settings [12]. OG-MARL provides a collection of offline datasets and baselines for fully cooperative MARL task, built on top of existing multi-agent environments. The benchmark includes datasets generated by different behavior policies and proposes metrics for evaluation offline MARL algorithms under varying dataset compositions [12]. OG-MARL thus takes an important step toward standardized offline evaluation in multi-agent learning, but its domains remain abstract and do not focus on multi-robot continuous-control tasks or integration with robotics frameworks.

Together, D4RL amd OG-MARL illustrate how curated datasets and shared evaluation protocols can

accelerate progress in offline RL and offline MARL. However, there is still no analogous benchmark that provides offline datasets for realistic multi-robot environments, aligned with the dynamics, sensing, and communication patterns encountered in robotic swarms. SwarmBench aims to fill this gap by pairing multi-robot MARL task with standardized offline datasets and evaluation procedures enabling researchers to study both online and offline learning under a common task specification and API.

4 Research Methodology

4.1 Overall Research Approach

The thesis adopts a more computational, simulation-based research approach. The primary goal is to design and implement a modular benchmarking framework for MARL in multi-robot environments, and to use this framework to conduct controlled experiments comparing representative MARL algorithms. The work consists of two tightly coupled components: (i) the software engineering and design of the SwarmBench framework, including its task specification, APIs, and simulation backends; and (ii) an empirical study that evaluated several MARL algorithms along axes relevant to deployment in multi-robot settings, such as offline versus online training, communication reliability, and observation fidelity. No human subjects are involved, and any experiments with physical robots will be conducted, if at all, under existing lab safety protocols.

4.2 Framework Design and Implementation

4.2.1 Core Architecture

SwarmBench will be implemented as a modular Python library, with a clear separation between task definitions, simulation backends, and evaluation utilities. At its core, the framework will define a *task specification* (TaskSpec) that described the abstract propertied of a multi-robot MARL task: the number and role of agents, the global and per-agent observation and action spaces, the reward structure, termination conditions, and the communication model. Agents will interact with tasks through a standardized multi-agent API inspired by existing libraries such as PettingZoo and Gym, exposing reset and step functions and returning observations, rewards, termination flags, and auxiliary information.

To support different simulation backends, SwarmBench will introduce a backend interface that implements the TaskSpec in concrete environments. Initial backends will focus on 2D continuous-control simulations suitable for multi-robot navigation and coordination. The backend interface will be designed so that future work can integrate more realistic robotics simulators or ROS-based environments without changing the TaskSpec or MARL algorithm code. Configuration files (e.g., in YAML or JSON) will be used to define task

parameters, communication settings, and observation modalities, facilitating reproducibility and systematic sweeps over experimental conditions.

4.2.2 Software Stack and Engineering Practices

The framework will be implemented in Python using commonly adopted deep learning and scientific computing libraries (e.g., Pytorch and NumPy). Where appropriate, vectorized operations will be used to simulate multiple parallel environment instances for efficient training. The codebase will be structured to separate environment logic, logging and dataset handling, and algorithm wrappers. Basic unit tests and example scripts will be provided to verify that tasks behave as expected and that evaluation pipelines can be reproduced. The final version of SwarmBench is intended to be released as open-source software, along with documentation and configuration files used in the experiments.

4.3 Benchmark Tasks

SwarmBench will initially include a small suite of cooperative multi-robot tasks designed to capture core challenges in multi-robot coordination while remaining computationally tractable. Each task will be specified through the TaskSpec and implemented in at least one simulation backend. The current plan it to develop:

- **Cooperative Navigation:** A set of robots must navigate from randomized start locations to assigned goals in a cluttered 2D environment while avoiding collisions with obstacles and each other. Observations will include each agent's position, velocity, and local information about nearby agents and obstacles. Actions will be continuous velocity of acceleration commands, and rewards will balance goal-reaching, time efficiency, and collision penalties.
- **Formation or Coordination Task:** A group of robots must maintain or achieve a target formation (e.g., line, circle, or convoy) while following a trajectory or avoiding obstacles. Observations will emphasize relative positions and velocities, and rewards will measure deviation from the desired formation and safety constraints.

For each task, the thesis fill precisely define the state and observation spaces, actions spaces, reward functions, and termination conditions. The tasks will be designed so that they can be instantiated under different communication and sensing configuration (e.g., with or without communication, low-dimensional versus richer observations) without changing their core structure. This enables the same task to be used for multiple experimental comparisons.

4.4 Algorithms and Training Regimes

The study will focus on a set fo representative MARL algorithms that cover the dominant families described in the literature review: actor-critic CTDE, value-decomposition, and multi-agent policy-gradient methods. Concretely, the plan is to include:

- At least one CTDE actor-critic method such as MADDPG or a similar centralized-critic architecture.
- A value-decomposition method such as VDN.
- A monotonic mixing-based method such as QMIX.
- An on-policy multi-agent policy-gradient method such as MAPPO.

Each algorithm will be implemented using a common interface so that they can be swapped in and out of the training pipeline with minimal changes to the surrounding code. Hyperparameters will be set based on values reported in the original papers or in widely used open-source implementations, with limited tuning to avoid overfitting to any single task. Training will be conducted under consistent conditions across algorithms, including fixed random seeds, episode limits, and total environment interaction budgets, to facilitate fair comparison.

Two primary training regimes will be considered:

- **Online MARL:** Algorithms interact with the environment directly, collecting experience as they learn. Thus regime reflects typical MARL benchmarks and provides a baseline for performance when exploration is allowed.
- **Offline MARL:** Algorithms are trained solely from fixed datasets generated beforehand by one or more behavior policies (e.g., partially trained policies, random policies, or simple hand-crafted controllers). Policies learned offline will then be evaluated in the same environments without further learning. This regime logged multi-robot data.

4.5 Offline Dataset Generation and Handling

To support offline MARL experiments, SwarmBench will include tooling to generate, store, and reuse standardized multi-agent datasets for each task. Datasets will consist of sequences of transitions, including per-agent observations, actions, rewards, termination flags, and any communication messages or global state variables required for analysis. Data will be collected by running a mixture of behavior policies.

Datasets will be saved in formats suitable for efficient loading and batching (e.g., compressed NumPy arrays or Pytorch tensors), along with metadata describing the behavior policies, dataset sizes, and task

configurations. The thesis will define a standard train/validation/test splits and evaluations protocols so that offline MARL results can be reproduced and extended into future work.

4.6 Communication and Observation Configurations

A key dimension of the benchmark is robustness to communication constraints and variations in observation fidelity. Swarmbench will therefore allow tasks to be instantiated under different communication and observation configurations via the TaskSpec and configuration files.

For communication, the framework will support configurable models including:

- No explicit communication (agents rely only on local observations).
- Reliable communication with fixed bandwidth, parameterized by user-specified probabilities or constraints.

These models will be implemented as wrappers around the underlying task, modifying what information agents can exchange and when. For observations, the framework will support at least two levels of fidelity:

- Low-dimensional state-based observations (e.g., positions, velocities, and relative distances).
- Richer, sensor-like observations that approximate range or bearing measurements or coarser occupancy information, which may be more representative of real robot sensing than perfect state information.

By systematically varying communication and observation setting while keeping the underlying task fixed, the study will be able to assess how sensitive different MARL algorithms are to realistic information constraints.

4.7 Evaluation Metrics and Analysis

Evaluation will focus on metrics that capture both task performance and safety, as well as robustness to changes in data, communication, and observation conditions. For each task and configuration, the following metrics are planned:

- **Task Performance:** Average episodic return, success rate (fraction of episodes in which all goals are reached or formation is maintained), and time-to-goal or time-to-completion.
- **Safety:** Number of inter-robot collisions, collisions with obstacles, or other constraint violations per episode.

- **Robustness:** Degradation in performance metrics as communication reliability if reduced, bandwidth is constrained, or observation fidelity is lowered; comparison of online versus offline training under matched evaluation conditions.

All metics will be estimated over multiple random seeds to account for stochasticity in initialization and environment dynamics. The results will be presented using learning curves, performance summaries with confidence intervals, and robustness plots that show performance as a function of communication or observation parameters. The analysis will focus on identifying which algorithm families are more robust to information constraints and data limitations, and on highlighting trade-offs between sample efficiency, final performance, and safety in multi-robot MARL.

4.8 Reproducibility and Deliverables

To support reproducibility, the thesis will document all task configurations, algorithm hyperparameters, dataset generation procedures, and evaluation scripts. The code for SwarBench, along with configuration files and a subset of the generated datasets, will be made available in a public repository where possible. The final deliverables will include (i) the SwarmBench framework itself, (ii) the benchmark task and offline datasets developed in this work, and (iii) a comprehensive experimental study of MARL algorithms on these tasks, providing baseline results and analysis that future researchers can build upon.

References

References

[1] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[2] P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls, and T. Graepel. Value-decomposition networks for cooperative multi-agent learning based on team reward. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, 2018.

[3] T. Rashid, M. Samvelyan, C. Schröder de Witt, G. Farquhar, J. N. Foerster, and S. Whiteson. QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018.

[4] C. Yu, A. Velu, E. Vinitsky, Y. Wang, A. M. Bayen, and Y. Wu. The surprising effectiveness of PPO in cooperative multi-agent games. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. (Also available as arXiv:2103.01955.)

[5] J. K. Terry, B. Black, N. Grammel, A. Hari, R. Sullivan, L. Santos, C. Wild, R. Perez-Vicente, A. Terry, C. Lawrence, K. Tu, N. L. Williams, and R. Laroche. PettingZoo: Gym for multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2021.

[6] J. Z. Leibo, E. Hughes, M. Lanctot, A. Gruslys, P. A. Ortega, T. Grau-Moya, T. Probabilistic, *et al.* Scalable evaluation of multi-agent reinforcement learning with Melting Pot. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, 2021.

[7] M. Bettini, M. Coppola, A. Prorok, and F. Farshidian. VMAS: A vectorized multi-agent simulator for collective robot learning. In *Distributed Autonomous Robotic Systems (DARS)*, 2022. (Extended version available as arXiv:2207.03530.)

[8] W. Chen, F. Kugele, Y. Wang, A. Albu-Schaeffer, and S. Stückler. MultiRoboLearn: An open-source framework for deep reinforcement learning with multiple robots. *arXiv preprint* arXiv:2209.13760, 2023.

[9] Z. Liang, J. Cao, J. Chen, and X. Liu. SMART: A scalable multi-robot reinforcement learning platform. *arXiv preprint*, 2022. (Under submission to IEEE Internet of Things Journal.)

[10] D. Ferigo, S. Traversaro, G. Metta, and D. Pucci. Gym-Ignition: Reproducible robotic simulations for reinforcement learning. In *2020 IEEE/SICE International Symposium on System Integration (SII)*, pages 885–890, 2020.

[11] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine. D4RL: Datasets for deep data-driven reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2021. (Also available as arXiv:2004.07219.)

[12] C. Formanek, A. Jeewa, J. Shock, and A. Pretorius. Off-the-Grid MARL: Datasets and baselines for offline multi-agent reinforcement learning. In *Proceedings of the 22nd International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, 2023. (Also available as arXiv:2302.00521.)

Dropbox Sign

Audit trail

Title	HUT Thesis Proposal
File name	MARLThesisProposal.pdf
Document ID	d13268433ea8b3e18463279b28e2179da1efad8d
Audit trail date format	MM / DD / YYYY
Status	● Signed

Document History

SENT	**12 / 03 / 2025** 15:48:48 UTC-3	Sent for signature to Chinwendu Enyioha (cenyioha@ucf.edu), Yunjun Xu (yunjun.xu@ucf.edu) and Manoj Chopra (manoj.chopra@ucf.edu) from ak565492@ucf.edu IP: 99.45.244.217
VIEWED	**12 / 03 / 2025** 16:21:58 UTC-3	Viewed by Manoj Chopra (manoj.chopra@ucf.edu) IP: 132.170.196.242
SIGNED	**12 / 03 / 2025** 16:22:12 UTC-3	Signed by Manoj Chopra (manoj.chopra@ucf.edu) IP: 132.170.196.242
VIEWED	**12 / 03 / 2025** 17:26:47 UTC-3	Viewed by Yunjun Xu (yunjun.xu@ucf.edu) IP: 108.253.161.195
SIGNED	**12 / 03 / 2025** 17:26:57 UTC-3	Signed by Yunjun Xu (yunjun.xu@ucf.edu) IP: 108.253.161.195

Dropbox Sign

Audit trail

Title	HUT Thesis Proposal
File name	MARLThesisProposal.pdf
Document ID	d13268433ea8b3e18463279b28e2179da1efad8d
Audit trail date format	MM / DD / YYYY
Status	Signed

Document History

VIEWED	**12 / 03 / 2025** 23:43:33 UTC-3	Viewed by Chinwendu Enyioha (cenyioha@ucf.edu) IP: 23.234.76.209
SIGNED	**12 / 03 / 2025** 23:44:07 UTC-3	Signed by Chinwendu Enyioha (cenyioha@ucf.edu) IP: 23.234.76.209
COMPLETED	**12 / 03 / 2025** 23:44:07 UTC-3	The document has been completed.