# Learning Modular Robotic Control via Reinforcement Learning using Attention based Global State Prediction

by

Brian Francis

A Thesis

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Master of Science

In

Robotics Engineering

by

August 2023

Approved:

Prof. Carlo Pinciroli, Advisor

PhD Candidate Josh Bloom, Advisor

Prof. Greg Lewin, Robotics Engineering

Prof. Cagdas Onal, Robotics Engineering

# Abstract

Deep Reinforcement Learning (DRL) has shown remarkable success in the control of single-robot applications. The approach has seen impressive results when applied to multi-robot coordination, but it has some notable shortcomings to overcome. Even though it is becoming increasingly popular for real-world multi-robot autonomy, DRL struggles as the complexity of the control system being developed increases. In systems with a high number of agents and consequently Degrees-of-Freedom (DOF), the training process can be prohibitively time-consuming or fraught with issues that make it difficult to learn optimal behaviors. One of the primary issues that DRL is faced with in multi-robot systems is managing the simultaneous learning process where the inter-agent interactions provide inconsistent information to the model. We investigated Attention-based Global State Prediction (AGSP) which uses information from neighbors to form a belief over the outcome of all the agents in order to overcome this instability in the training process. AGSP is able to predict future states accurately even over a large number of agents using information communicated about the collective actions. We used AGSP in a decentralized modular locomotion task and empirically evaluated the emergent properties. We found that AGSP produces policies that exhibit superior stability and adaptability. This makes AGSP a useful tool for developing safe and consistent controllers with low rates of failure.

# Acknowledgements

I am profoundly grateful to the countless individuals who have contributed to the realization of this endeavor, both through their unwavering support and invaluable guidance. This project would not have been possible without the steadfast encouragement and love of my family, who have been my constant pillars of strength. Your belief in me has been a driving force, and for that, I am eternally thankful.

I extend my deepest appreciation to my advisors, Professor Carlo Pinciroli and PhD Josh Bloom, for their exceptional mentorship, dedication, and scholarly insights. You both put tremendous effort into my education at a time in my academics where I needed it and completely changed my trajectory. Your guidance has been instrumental in shaping my understanding, fostering my intellectual growth, and steering me towards the path of excellence. Your willingness to share your expertise and engage in discussion has been pivotal in refining my understanding of the field, and your contributions have undoubtedly enhanced the quality and depth of this work.

I would like to express my gratitude to the academic community of WPI for providing me with an environment conducive to learning and exploration. The interactions with peers and colleagues have enriched my perspective and broadened my horizons, and this campus has meant the world to me in my time here. Special thanks are also due to the rest of the NESTLab, who have generously shared their knowledge, time, and resources, thereby enriching the tapestry of this project.

In conclusion, this work stands as a testament to the collective effort of all those who have contributed to its realization. Nobody can complete something like this alone, and your support and contributions have left an indelible mark on both my academic journey and personal growth. As I move forward, I carry with me the lessons, insights, and connections that have been forged during this endeavor.

With sincere gratitude,

Brian Francis

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Modular robots (MR) are robotic systems that consist of individual modules where a module is generally an actuator that enables locomotion or shape-changing (Singh et al., 2022). These modules are interchangeable either autonomously or physical connection allowing for variable behavior, MR have experienced a remarkable expansion in its applications, leading to the development of increasingly diverse and complex systems (Alattas et al., 2019). This tremendous growth can be attributed to the flexible nature of these robotic systems, which permits the assembly and reconfiguration of individual modules to create various robot configurations tailored to different tasks. This modularity provides significant advantages, such as adaptability and scalability, allowing MR to excel in a wide range of applications from manufacturing to exploration and even medical interventions. (Grabowski et al., 2000) (Ding et al., 2022)

However, as the complexity of MR systems grows, traditional control methods encounter significant challenges, particularly concerning the number of parallel degrees-of-freedom (DoFs) (Varshavskaya et al., 2008). In MR, each module typically possesses its own degrees of freedom, granting it the capability to move or execute actions independently. As more and more modules are integrated into the system, the number of parallel DoFs increases exponentially, leading to a higher level of intricacy in the control structure. The surge in parallel DoFs introduces issues of coordination, synchronization, and overall system stability. The complexity of managing multiple modules operating simultaneously demands advanced control algorithms and computational power~~.~~ (OpenAI et al., 2019). Traditional control methods, designed for more conventional robotic systems with limited DoFs, struggle to cope with the dynamic and non-linear behaviors exhibited by these highly interconnected MR systems (Whitman et al., 2021).

While MR are highly editable, even if you can develop a methodology to control one configuration, once you change the setup you have to devise and tune a new policy. This adds to the complexity and burden of controlling MR systems effectively. To simplify the development of control policies for MR systems, reinforcement learning has emerged as a successful approach that is capable of handling a large number of modules (Ahmadzadeh & Masehian, 2015). With reinforcement learning, control policies for MR systems can be rapidly developed, as the agent can learn through trial and error and optimize its actions based on the received feedback. This approach can help in mitigating the challenges associated with the complexity and editability of MR systems. However, single-agent approaches are hampered by non-stationarity caused from several modules learning simultaneously (Busoniu et al., 2008).

To address these challenges, researchers and engineers have been exploring novel control strategies, such as bio-inspired algorithms, swarm intelligence, and distributed control systems (Schilling et al., 2021). By treating each module as an individually controllable agent, it is possible to study the collective emergent behavior that arises from the interaction of these components. These approaches have seen success by dividing up the computation to allow for control methodologies to scale to larger numbers of modules working together. These innovative approaches leverage the self-organization and emergent behaviors observed in natural systems, which can be highly advantageous in managing large-scale MR configurations (Thor & Manoonpong, 2022). By incorporating these cutting-edge techniques, we aim to enhance the adaptability and autonomy of modular robotic systems, unlocking their full potential for complex tasks and environments.

## 1.1 Problem Statement

The primary objective of this research is to train a single decentralized model capable of achieving successful locomotion when applied to a distributed modular robotic platform with incomplete information. The focus of our investigation is on MR in the lattice configurations where the modules are not attached sequentially but rather affixed to another mechanical link. As a result, their control becomes inherently interconnected, as they exert forces on each other, impacting the overall motion. However, not every motion necessarily has an impact on the others leading to inconsistent feedback from the system. This complicates the learning process as the connection between modules introduces non-stationarity into the system where the optimal policy moves as a result of the inter-module interactions being different each iteration. Since the system is learning and updating its policies each episode the correlation between actions and outcomes is difficult for the model to track.

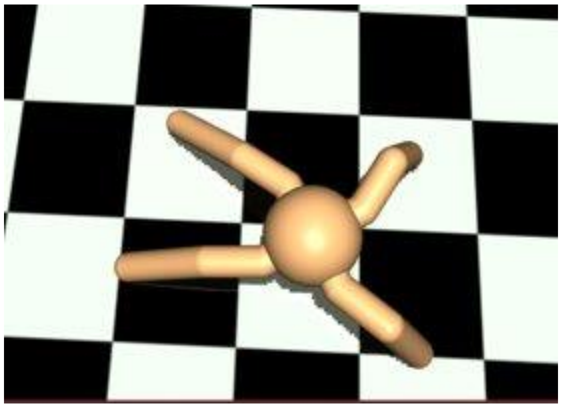

*Figure 1: MuJoCo Ant Environment*

To investigate this problem, we focused on a specific modular locomotion task called the ant (Figure 1), which is part of the OpenAI Gym Multi-Joint Dynamics with Contact (MuJoCo) library (Todorov et al., 2012). In this task a set of legs are attached to a central body and tasked with travelling in the positive X direction as quickly as possible. The reward policy promotes positive motion while penalizing excessive energy expenditure. This task structure aims to develop a policy that not only encompasses an effective gait but also optimizes it for maximum efficiency. Each leg consists of a “hip” where it attaches to the body and a “knee” joint in the middle of the leg. We divide the ant up into several independent elements where each leg is a single module with control over its own hip and knee joint, and the module has observability over information from other legs in the system from a simulated sensor array including positional and joint information. To see success all the legs must work together in a coordinated approach that not only moves the body but does it in such a way that it can maintain momentum consistently.

Our primary goal was to develop a robust and efficient locomotion policy that enables modules to cooperate effectively and exploit their physical connection to expand their set of observations. Our approach aimed to go beyond relying solely on communication between modules and instead leverage their interconnectivity to enhance coordination and adaptability. By addressing these challenges, we aimed to contribute to the advancement of decentralized control methods for modular robotic systems, empowering them with greater adaptability and autonomy in navigating complex scenarios. Our research aimed to shed light on novel solutions that harness the intrinsic relationships and physical connections within the MR system, ultimately paving the way for more efficient and capable robots.

## 1.2 Contributions

We demonstrate that the individual components of an MR can learn adaptable policies using Reinforcement Learning (RL) without learning distinct models for each module. This accelerates the development of MR systems, making them potentially more versatile and applicable across a wide range of tasks and environments. We investigate Attention-based Global State Prediction (AGSP) which not only fosters the creation of successful control policies but also ensures consistent and reliable performance. AGSP, an extension from Global State Prediction (Bloom et al., 2023), leverages attention mechanisms to predict and maintain a comprehensive global state representation, facilitating effective inter-module coordination even amidst non-stationarity and intricate interactions. By introducing AGSP to the learning process, we mitigate the limitations of traditional RL methods, enhancing adaptability and improving policy performance.

To evaluate the effectiveness of AGSP, we conducted a comprehensive comparative analysis of the various training approaches. This includes policies trained under centralized and distributed control settings, as well as the integration of AGSP. The results demonstrate the advantages of AGSP in achieving improved control performance and consistency within MR systems particularly as complexity increases, showcasing its potential as a powerful tool for overcoming challenges associated with coordination and non-stationarity.

By addressing pressing challenges in MR, specifically control complexity and generalization, through the introduction of innovative techniques such as AGSP, our research contributes valuable insights towards enhancing the robustness and adaptability of MR systems. This advancement holds the potential to drive more efficient and capable robotic solutions in real-world applications. Through empirical evaluation, comparisons, and innovative contributions, this research fosters a deeper understanding of how MR systems can achieve enhanced adaptability, coordination, and performance, ultimately paving the way for more resilient and versatile robotic solutions.

# Chapter 2

# Background and Related Work

## 2.1 Reinforcement Learning (RL)

Reinforcement Learning (RL) is a branch of machine learning concerned with training intelligent agents to make decisions and take actions in an environment to achieve specific goals. Unlike supervised learning, where the model is provided with labeled data, or unsupervised learning, where the model tries to identify patterns without explicit guidance, reinforcement learning relies on the concept of an agent interacting with an environment, learning from feedback signals, and optimizing its behavior over time.

At the core of RL lies the Markov Decision Process (MDP), a mathematical framework used to model sequential decision-making problems under uncertainty. The MDP provides a formal way to represent the essential elements of an RL problem and is defined by a tuple ⟨S, A, R, P⟩. At each time step the agent is in state s ∈ S where S is the set of all possible states and chooses an action a ∈ A where A is the set of all possible actions. The Transition Model represented by P(s'| s, a) describes the probability of moving to state s' from state s when the agent takes action a. The decision is evaluated based on the reward, a feedback signal that indicates how well the agent is performing with respect to its goal defined as R(s,a,s′):S×A×S→ℝ. The objective of the agent is to maximize the cumulative reward over time. The typical approach to solving an MDP is to use the Bellman equation that introduces a value to each state-action pair of Q(s,a). This is a valuation on the expected future rewards that is modified by γ the discount factor, and the agent chooses to maximize the value gained with each action. If there are a relatively small number of possible state action pairs Q can take the form of a table or array or in more complex environments, it is approximated by a neural network.

$$Q(s,a) = \Sigma_{s'} P(s'|s,a)\big(R(s,a,s') + \gamma \max Q\,(s',a')\big)$$

The MDP has an assumption that the state is a complete picture of task relevant information which is not always true particularly when considering a robotics application that will be limited to the information available from a conventional sensor array. Because there is an incomplete perception of the environment these applications become a Partially Observable Markov Decision Process (POMDP) that is defined by the tuple ⟨S, A, R, P, **Ω**, O⟩. This change introduces the observation $o \in \Omega(s)$ that is a partial observation of the true state s according to the function O(o |s', a).

If a S and A are small enough, then the value of every state-action pair can be mapped to a table to simplify decision making. However, as RL applications have expanded to more complex and high-dimensional state spaces, traditional tabular methods became infeasible. As the number of possible state-action pairs increases it becomes increasingly inefficient to try to visit and test every combination. Instead, approximate Reinforcement Learning techniques emerged to address this challenge. One of the most significant breakthroughs in RL came with the advent of Deep Reinforcement Learning (DRL). The introduction of Deep Q-Networks (DQN) (Mnih et al., 2015) combined Q-learning with deep neural networks to handle high-dimensional state spaces and achieved human-level performance on Atari 2600 games. This approach was limited and unable to handle large state and action spaces or continuous environments. Another approach that designed to handle these limitations came in the form of Deterministic Policy Gradient (DPG) (Silver et al., 2014) that worked by directly optimizing the policy parameters utilizing an actor-critic method to maximize the expected cumulative reward. Both DPG and DQN were further developed with the introduction Deep Deterministic Policy Gradient (DDPG) (Lillicrap et al., 2019) which learned both Q-function and policy. This approach was highly successful in continuous action spaces, making it applicable to tasks in robotics and control.

Building upon the success of DDPG, the seminal work (Fujimoto et al., 2018) proposed TD3 as a variant of DDPG with three key innovations that sought to address the issues of overestimation bias and exploration noise. First, TD3 employs a pair of critics to reduce value function overestimation, leading to more stable and accurate value estimates. Second, it introduces target policy smoothing to address the problem of policy overoptimism. By adding random noise to target policy actions, TD3 encourages conservative policy updates, improving the policy's robustness. Lastly, TD3 incorporates a delayed policy update mechanism, reducing the variance of the value estimates and further stabilizing the learning process. TD3 has demonstrated superior sample efficiency and learning stability on a wide range of continuous control tasks, making it a popular choice for researchers in RL.

The vanishing gradient problem is a challenge in deep learning where the gradients of the loss function become extremely small as they are backpropagated through multiple layers of a neural network during training. This phenomenon can lead to slow or stalled learning, as the updates to the network's weights become too tiny to effectively adjust the model's parameters, resulting in slower convergence and degraded performance. To overcome the vanishing gradient problem and support the learning of sequenced data recent RL works have utilized the transformer model (Liu et al., 2020). While originally designed for natural language processing tasks (Vaswani et al., 2023), transformers have been successfully adapted to various domains. Transformers use self-attention which computes weighted representations of input elements by considering their relationships with other elements in the same sequence. This model captures dependencies between different elements, usually time series, within the state or observation space. The self-attention mechanism in Transformers allows the model to attend to different parts of the input sequence and capture dependencies between elements, making it highly effective for handling long-range dependencies in sequential data. By modeling these dependencies this approach offers improved performance and interpretability in Reinforcement Learning settings. (Vaswani et al., 2023)

## 2.2 Multi-Agent Reinforcement Learning (MARL)

Multi-Agent Reinforcement Learning (MARL) has emerged as an area of research focusing on scenarios where multiple agents interact within a shared environment, aiming to optimize their individual objectives while influencing the overall system behavior. This can be further extended from the POMDP to the Decentralized Partially Observable Markov Decision Process (Dec-POMDP). Instead of a single agent, there are a set of agents I in the environment,

$A=\times_i A_i$ is the set of joint actions, and $\Omega=\times_i \Omega_i$ is the set of joint observations. The representation is denoted by the updated tuple $\langle I, S, \{A_i\}, R, P, \{\Omega_i\}, O\rangle$.

The introduction of other agents who are learning and interacting with each other simultaneously introduces the issue of non-stationarity, also called the moving target problem, which causes issues in policy convergence. The difficulty in training arises from the optimal policy for the agents moving as they learn and update their policies. Due to the simultaneous updates the inter-agent interactions are different depending on their current policy leading to inconsistent experience. Due to the prevalence of the issue, there is a body of work dedicated to attempting to mitigate non-stationarity by improving coordination, or observation. (Nayak et al., 2023) (Zhou et al., 2022) (Chalkiadakis & Boutilier, 2003).

One such way to combat non-stationarity is to allow for communication between agents. Communication takes a variety of different forms in RL from direct communication to shared gradient values. (Foerster et al., 2016) proposed a differentiable communication channel to enable agents to learn to communicate. Building upon the actor-critic framework, (Lowe et al., 2020) introduced the multi-agent deep deterministic policy gradient (MADDPG) algorithm that enabled agents to learn individual policies and value functions while considering the actions of other agents with a centralized critic. (Sukhbaatar et al., 2018) introduced role-based dialogues where agents are assigned specific roles, and they learn to communicate effectively to achieve their joint objectives. The dialogue-based communication enables agents to exchange information and coordinate their actions efficiently, leading to improved coordination and task performance in cooperative multi-agent scenarios.

Another approach to tackling non-stationarity involves the use of a belief state or prediction of other agent’s behavior. (Rodríguez et al., 1999) proposed maintaining approximating belief propagation in POMDPs. (Zhang et al., 2022) showed that belief can benefit the performance of multiagent systems. By maintaining beliefs about the hidden states of the environment, agents can reason about the potential changes in their surroundings, leading to more informed decision-making and improved coordination in complex and uncertain environments. (Wu et al., 2021) introduced a spatial intention map that added a predicted trajectory of other agent’s movements that improved spatial coordination between agents. This prediction can be used to plan tasks or designate roles (Stulp et al., 2006). Global State

Prediction (GSP) makes use of the inherent properties of physically connected systems to learn a predictive methodology for collective transport (Bloom et al., 2023).

Despite notable progress, challenges persist in MARL, including the scalability of algorithms to handle large agent populations, stability in learning, and avoiding undesirable emergent behaviors.

## 2.3 Modular locomotion

Modular robotic systems (MRS) are a class of robotic systems characterized by their ability to dynamically change their shape in response to the task and environmental conditions, achieved through adopting various morphologies. These versatile robots are composed of multiple modules, each equipped with a limited number of degrees of freedom (DOFs). The modules are designed with connection mechanisms that allow them to cooperatively connect or detach from one another, enabling the formation of aggregate structures and shapes sometimes autonomously. MRS offer the benefit of flexibility in application due to this customizability. However, developing control structures for MRS presents a difficult challenge due to the number of possible configurations that can be tedious and time consuming to produce via traditional methods (Whitman et al., 2021).

These methods for legged locomotion largely make use of central pattern generators (CPG) (Fukuoka et al., 2015) in order to develop a gait cycle for a specific robot. These CPGs are based on biological processes that generate rhythmic functions such as breathing or walking (Marder & Bucher, 2001). They form a loop that takes in sensory information and ties it to output mechanisms, such as motors in the case of robotics. CPGs can adapt to changes in the environment or the organism's needs. For instance, when walking on uneven terrain, sensory feedback can modify the CPG's output to adjust the walking pattern for stability (McKenna & Zeltzer, 1990). Classic gait generation methods have been used from small bipedal systems (Khan & Mandava, 2023) to large scale systems including eight-legged robotic platforms (Grzelczyk & Awrejcewicz, 2019). The use of more advanced body mechanisms including passive spines have allowed for the development of gaits in systems of even larger sizes such as (Tang et al., 2022) that developed an undulating gait for a robot with any even number of legs.

However, a notable shortcoming of these methods is their limited scalability and adaptability when faced with dynamic systems such as modular robots. While CPGs offer a foundation for gait generation and adaptation, they are limiting when designing modular systems due to the large number of possible configurations that can arise from even a small set of modules (Whitman et al., 2021). Reinforcement learning has been used to streamline the process of developing controllers for modular locomotion due to the ease of development (Schaff et al., 2018), and as such is much more time effective when designing multiple controllers for the variety of configurations of a modular system. The task of training a locomotion controller is not trivial, particularly for legged locomotion, and there are a variety of approaches for training. Hierarchical control is a biologically inspired approach where different models are trained on different elements of the locomotion process including foot placement, path planning, and leg movement as examples (Amri et al., 2023). This approach often uses a central planning brain that designates operations and switches between different behaviors. Hwangbo et al. (2019) demonstrate that, in a matter of hours, this approach develops a policy that outperforms existing controllers.

While hierarchical control has proven effective in achieving specialized control for specific morphologies, it lacks generalizability and transferability to other systems, necessitating significant restructuring to be applied elsewhere. Furthermore, hierarchical control approaches struggle to adapt behavioral primitives learned in one context to new and varying situations (Schilling et al., 2020). In response to these limitations, recent advances in developing more generalized learning processes have led to a shift towards decentralized control methods, where each individual module within the robotic system is equipped with its own controller. By adopting this decentralized approach, training offers notable advantages, such as increased robustness to noise and faster convergence speeds, thanks to the utilization of smaller, more manageable models (Schilling et al., 2021). These benefits make decentralized control a highly reliable and efficient means to rapidly learn locomotion models for robots. This decentralized paradigm not only enhances adaptability across different contexts but also lays the foundation for more scalable and adaptable robotic systems capable of handling diverse and challenging tasks effectively.

# Chapter 3

# Methodology

## 3.1 Task Formulation

This study aimed to evaluate the effectiveness of RL training methodologies on a modular, legged locomotion task from the OpenAI Gym MuJoCo library (Schulman et al., 2018). The task could be completed without the use of RL, but our goal was to develop an end-to-end RL methodology to improve the development of RL controllers. The primary objective was to assess the capacity of various methods to promote cooperation and facilitate proficient multi-agent policy learning. The legged locomotion system was designed with a modular arrangement, wherein each leg was symmetrically positioned around a central hub. These legs, while essentially identical, differed only in their spatial placement and the four-legged configuration is shown in Figure 2.

Each leg consisted of two distinct segments connected by two joints. The initial joint, termed the "hip," functioned as a revolute joint linking the leg to the central hub of the system. This joint enabled rotational movement around the z-axis, with a range of up to $\pm$30 degrees. The second joint, referred to as the "knee," is a revolute joint responsible for connecting the lower leg to the upper leg. The knee joint facilitated rotation perpendicular to the upper leg from in a range from 30 - 70 degrees.

In the original, unmodified environment, the entire system is treated as a unified entity, sharing observations and actions. The action space is comprised of eight actions all pertaining to the control of the leg joints. Positive or negative torque on a continuous range of (-1,1) can be applied to all four hip joints and all four knee joints at each time step. The observation space by default is an array with 111 components. The observations include the Z coordinate of the body, the 4 quaternion components of the body orientation, the hip and then knee angle for each leg, the three-dimensional cartesian and rotational velocities of the body, the angular velocity for the hip and knee joints of each leg, and finally the contact forces on each individual body part which comprise the final 84 observations.

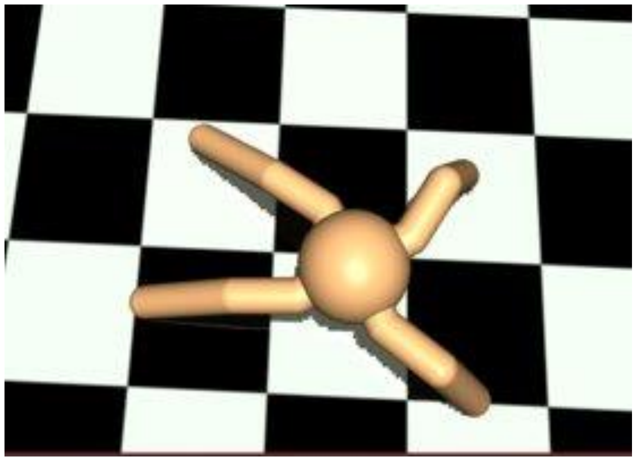

*Figure 2:Four-legged Ant robot in MuJoCo environment*

For the specific objectives of this research, a modular approach was adopted. Control was partitioned into discrete modules, each comprised of a single leg containing one hip and one knee joint.  This modularization aimed to facilitate a more detailed analysis of system interactions and learning strategies. Figure 3 shows the division for a four-legged configuration. However, the experimentation encompassed a broader scope, including an eight- legged configuration. In the eight- legged setup, additional legs were positioned within the gaps between the original four legs (Fig 4), aligned along the cardinal directions. The division followed the same pattern with each leg serving as one module. Other works have looked at having one agent control multiple modules (Peng et al., 2021), but we elected to have each agent only control one module to simplify the transition from between the four and eight leg configurations.

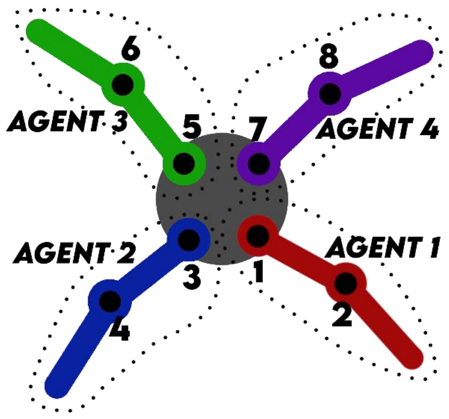

*Figure 3: Diagram showing the arrangement of agents in four-legged configuration*

Since the control of the system is distributed into separate modules, they all required their own set of observations. To make each module interchangeable but without having identical actions, the observation space was based on the position of the module (Table 1). The observations were structured starting with the cartesian position and velocity as well as the orientation of the central hub or "body" of the ant at the current time step. This information is identical for each module and in the same position at the start of the observation array. Following the global reference frame data, the observation sequence transitioned to the details of each leg including relative joint angles to the body, the angles between the upper and lower leg, and the instantaneous joint velocities. The information available to each module is the same, but the policy across the modules is asymmetrical depending on the placement of the leg. To ensure that each module could intuitively identify its relative placement, the leg observations were communicated in a clockwise rotation. This design choice endowed each module with the innate ability to implicitly perceive its position in relation to the other modules and their respective limbs. Without this each module would receive the same observations and therefore choose the same actions. This systematic rotation enhances the module's spatial awareness and contributed to their capacity for effective coordination and cooperative behavior that would not have been possible otherwise.

We wanted our observation space to be closely aligned with the types of sensors that might be practically available in a real-world scenario, therefore a conscious decision was made to exclude the consideration of contact forces within the observation space. This approach not only facilitated more realistic sensor modeling but also expedited the training process by reducing the overall size of the observational input.

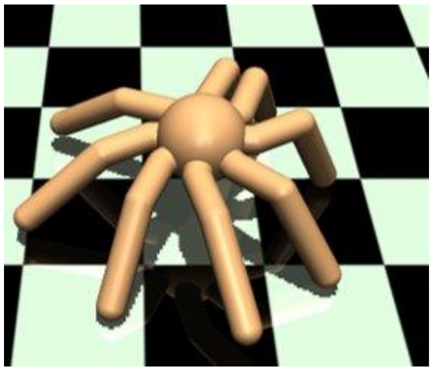

*Figure 4: Eight-legged Ant robot in MuJoCo environment*

## 3.2 Training

To optimize both the training process and the ultimate model scalability, we employed the approach of Aggregate Centralized Training with Decentralized Execution (A-CTDE) (Bloom et al., 2023). Each module's experiences were pooled together to make a single comprehensive replay buffer. This collective buffer not only reached saturation more swiftly but also encompassed a diverse array of experiences, enriching the training dataset (Egorov & Shpilman, 2022). The modules all independently interact with the environment, but their experiences are collected and used to train a single shared model. Consolidating experiences from every module and then training a shared model with each of their perspectives ensures that the model has a broader range of experiences than if it was trained on only a single module. During the experimental phase, the trained model was distributed to each module, allowing them to operate independently. The policy is applied to individual legs without any centralized control allowing the approach to be conducive for training large modular systems because the scalability issues of other training methodologies are dampened. The uniformity of observation and action spaces across modules streamlined the training process while still retaining the capacity for successful runtime operations through implicit differentiation.

*Table 1: Individual Module Observation Array*

| Number of Observations | Observations Included |
|---|---|
| 11 | Body observations ($O_b$) |
| 4 | Module n observations ($O_{li}$) |
| 4 | Module n+1 observations ($O_{li+1}$) |
| 4 | Module n+2 observations ($O_{li+2}$) |
| 4 | Module n+3 observations ($O_{li+(N-1)}$) |
| 3 | Predicted ΔV |

Given that the overarching objective of the task was to learn a dynamic gait cycle, the desired policy is dynamic and asymmetrical because different actions must be taken across the modules at any given time step. The imperative challenge for the model was to master an asymmetrical but cyclic approach to locomotion to both achieve balance and propel forward movement (Kelly & Murray, 1995). To facilitate this nuanced learning process, the observation space had to be structured in a specific manner. This configuration enabled each module to discern its unique identity within the swarm. Each module receives first the observations associated with the body of the ant ($O_b$) and then the leg observations ($O_{li}$) going clockwise around the body starting with itself. [$O_b$, $O_{li}$, $O_{li+1}$, … $O_{li+N-1}$]. This format allows each leg to develop a policy based on its position which allows for each leg to specialize in a different aspect of the gait cycle.

The reward structure is designed to complement the ant environment and is provided by the gym library. There are four elements that make up the reward equation, two positive factors and two negative factors. The primary reward is related to the motion of the body that rewards the system for completing the task of moving in the positive X direction and is defined as $R_f=\Delta X/d_t$ where $d_t$ is the frame skip parameter multiplied by frame time which as a default is equal to .05. The second positive reward is for keeping the robot healthy. Each time step the episode is not terminated the agent receives a reward of $R_h=1.0$ to learn a policy that avoids flipping or other unsafe actions. To promote efficient policies, the agent suffers a negative reward for its actions equal to the control cost ($C_1$) multiplied by the sum of the actions squared or $R_c= C_1*\text{sum}(A^2)$. The last element of the reward is a penalty for large contact forces that has a similar format as the action penalty but instead with the summation of contact forces ($F_c$) that takes the form of $R_{Fc} = C_2 * sum(clip(F_c\{F_c min, F_c max\})^2)$ . The cumulative reward equation is R= $R_f$+$R_h$+$R_c$+$R_{cf}$. When training the system with more legs, we normalized the value of $R_c$ since the increased number of legs causes the summation to out scale the other components of the reward equation. For such cases the control cost takes the form $R_c = C_1 * \frac{sum(A^2)}{N}$

The AGSP model addresses a significant challenge posed by the limited information available to each module within the environment. Specifically, in the absence of the AGSP model, the inherent lack of access to other modules' actions or the upcoming joint changes and acceleration in the system hampers a module's ability to effectively correlate its actions with the ensuing outcomes. This predicament stems from the absence of regular correlations between actions and results, making it arduous for modules to discern causal relationships.

AGSP by appending an array (P) which includes the changes in velocity along the three Cartesian components (ΔX, ΔY, ΔZ). The AGSP model, equipped with the same positional and joint data as the action model, but supplemented with access to the collective actions, undertakes the task of learning a comprehensive kinematic model of the system. The underlying objective is to predict how the entire body will react based on the actions of the system. This predictive capability serves as a bridge that empowers modules to understand the broader consequences of their actions, thereby enhancing their ability to make informed decisions in the dynamic environment.

The goal is to learn an approximate kinematic model of the system that can predict how the body will move based on the collective actions of the system. It is rewarded by the proportional accuracy of this prediction with the reward being defined by the error between the prediction P and the actual change in velocity ΔV normalized by the current velocity such that $R=[|(P_x-V_x)|+|(P_y-\Delta V_y)|+|(P_z-\Delta V_z)|]/\Delta V$. This reward structure is designed to promote granular accuracy that can detect both large spikes in velocity while not overestimating when the system is stationary or moving slowly.

AGSP is ingrained within the training process of the walking network. It operates in tandem with the walking policy learning, but it has a deliberate update delay. This approach prevents a detrimental reciprocal impact between the two policies. The AGSP model is updated every 5000 learning steps of the walking policy, thus ensuring an offset timeline that allows the walking policy to stabilize and integrate the predicted values into its learning before the AGSP model receives its subsequent update. However, to ensure that the prediction model is not lagging behind it performs 500 updates per iteration. This orchestration maintains a synchronized and harmonious advancement of both policies, optimizing the overall training process and culminating in a more comprehensive and effective multi-agent learning framework.

## 3.3 Models

For training the walking model, we employed the TD3 algorithm as it offers several key changes that improve performance over other learning algorithms as well as having been proven to be highly effective on the baseline task (Fujimoto et al., 2018). The structure of the model used in this work is shown in Figure 3. The walking model hyper-parameters were based off of the anecdotal best results from the seminal paper on the task (Fujimoto et al., 2018). These parameters were fixed for all testing throughout this work and were found to still be largely the best configuration for our revised experimentation with some revision. Hyper-parameters for the prediction model were found to be largely inconsequential to change although the configuration used produced the most accurate and consistent results.

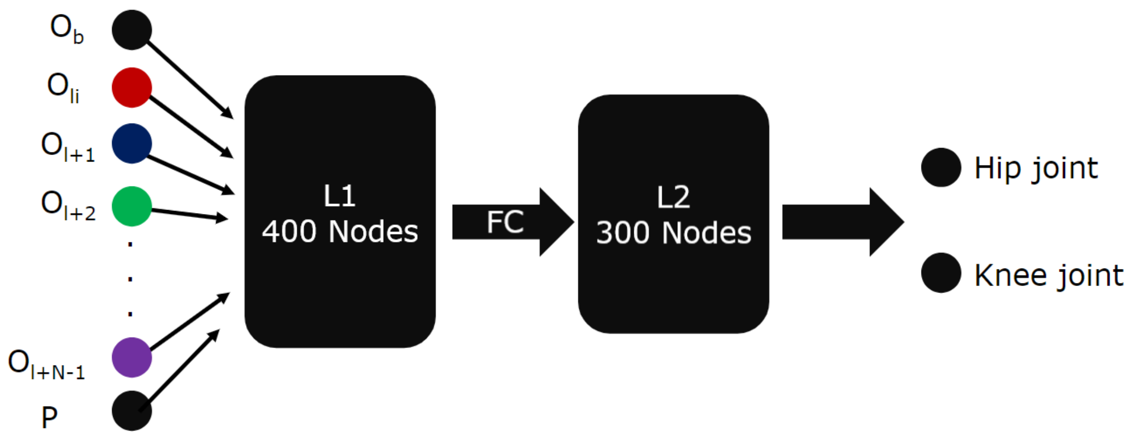

*Figure 5: Network diagram for TD3*

The decision to set the learning rates α at $10^{-3}$ for the actor and β at 2* $10^{-3}$ for the critic was based on a goal of finding an equilibrium between learning efficiency and model stability during the training process. This specific learning rate provides a calibrated compromise that allowed the model to swiftly learn from experiences while avoiding the pitfalls of overly aggressive updates that could lead to oscillations or overshooting. The selection of a batch size of 200 was chosen to strike an optimal balance between computational efficiency and gradient stability throughout the training regimen. This size enables the model to capture diverse experiences while mitigating the noise inherent in smaller batch sizes. The decision to adopt a discount factor (γ) of 0.99 prioritized total episode reward without excessively diminishing the significance of immediate gratification. This γ allowed for policy development that valued total acceleration balanced with the desire for continuous motion.

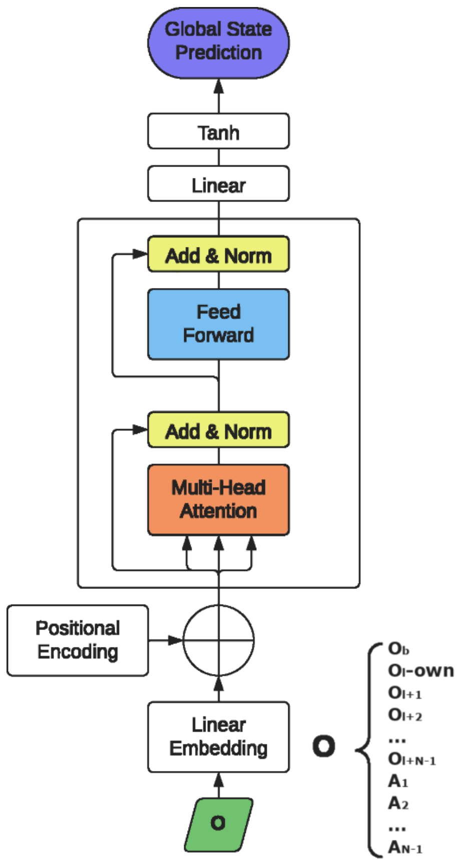

*Figure 6: Attention network model, Josh Bloom 2023 adapted with permission*

The AGSP network is a Transformer utilizing self-attention. The self-attention mechanism allows the model to attend to relevant information across different time steps and agent states, capturing long-range dependencies and intricate patterns crucial for accurately predicting complex kinematics. Furthermore, the parallel nature of self-attention operations facilitates efficient computation, making Transformers particularly suitable for distributed and hardware-accelerated implementations such as decentralized modular applications. The format and input information for the transformer are shown in Figure 4.

We chose to include eight layers in the transformer to allow the model to capture intricate hierarchical features and relationships within the data, thereby enhancing its capacity to learn the complex patterns and representations associated with kinematic data. With eight attention heads, the model exhibits a rich capacity to attend to diverse aspects of input data simultaneously. This promotes robust feature extraction and interaction modeling, enabling the network to comprehensively capture correlations among different elements of the input. Employing a forward expansion factor of four enhances the expressiveness of the model's transformations between its layers. This expansion contributes to the model's ability to learn intricate mappings, facilitating the capture of complex and nonlinear relationships present in the data. No neurons are excluded during training. This choice comes from a preference for model stability over regularization, allowing the model to fully exploit the available data without introducing the perturbations associated with dropout. The minimum-maximum action range binds the model's action outputs within a range of [-3, 3]. This normalization constrains the module's actions to a manageable and meaningful scale inside the realm of possibility for this task, promoting better convergence and facilitating smoother learning dynamics for the action policy.  By setting a maximum sequence length at five, the model processes sequences of limited temporal extent. This choice reflects a focus on capturing short-term dependencies and aligns with the assumption that critical interactions and dynamics occur within this temporal window, optimizing the model's performance for tasks with a limited time horizon.

# Chapter 4

# Evaluation

## 4.1 Experimental Setup

Experimentation was run on a computer equipped with an NVIDIA 960M graphics card using the Nvidia CUDA library as well as on a variety of computing nodes from the WPI Academic & Research Computing group that varied in strength but were limited to two Nvidia K20 GPUs. The OpenAI Gym API for reinforcement learning utilizing the integrated MuJoCo physics engine. The results for smaller swarm size are gathered using the default “Ant” baseline environment, and larger swarms utilize the baseline environment as a foundation with edited XML files to facilitate more advanced systems.  The episode can terminate before the time step cap if the body of the “Ant” leaves the “healthy” range of z values. In order to compare results with previous literature the v2 environment was used.

We used the PyTorch machine learning framework to design and train our models. The testing period per episode is the default value of 1000 time steps, and the training is confined to epochs of 1000 episodes before it begins training from a fresh model. The models that performed the best were saved if their total episode reward surpassed a value of 1500 for both the four-legged and eight-legged configurations. Any epoch that did not produce a value above the threshold was considered to not have converged to a successful policy and was classified as a failure.

## 4.2 Analysis

When undertaking result comparisons, our assessment framework encompassed five key characteristics, each integral in determining the success and efficacy of the various approaches.

The first is the task success directly considering the episode reward. If an approach was able to develop a policy capable of being highly rewarded that would be considered a positive outcome for a training episode. Along with this we monitored the training speed both relative to an epoch and in real time. If the model was able to converge to a successful policy in fewer experiences that would be considered superior, but we also considered if we were able to complete more training in a real time sense that would be beneficial as well.

After training, on episodes that had achieved a minimally successful policy, we also wanted to test the robustness of the policy and measure how consistently it could succeed in testing from a randomized start. Stability in learned policies was deemed paramount, ensuring that the training methodologies fostered the acquisition of safe and reliable behaviors for practical implementation in real-world contexts. For AGSP we additionally examined the accuracy of the predictions. This entailed an exploration of how effectively AGSP estimated state values, with an emphasis on how prediction accuracy influenced overall task outcomes. Furthermore, a granular examination of each element comprising the reward structure was conducted to discern nuanced behavioral disparities among policies trained through different approaches. It is pertinent to note that the dynamic nature of reinforcement learning permits the generation of diverse policies through repeated training iterations. To ensure fairness and accuracy in our comparisons, all analyses were confined to the most successful policies for each of the aforementioned characteristics compiled over nine training periods.

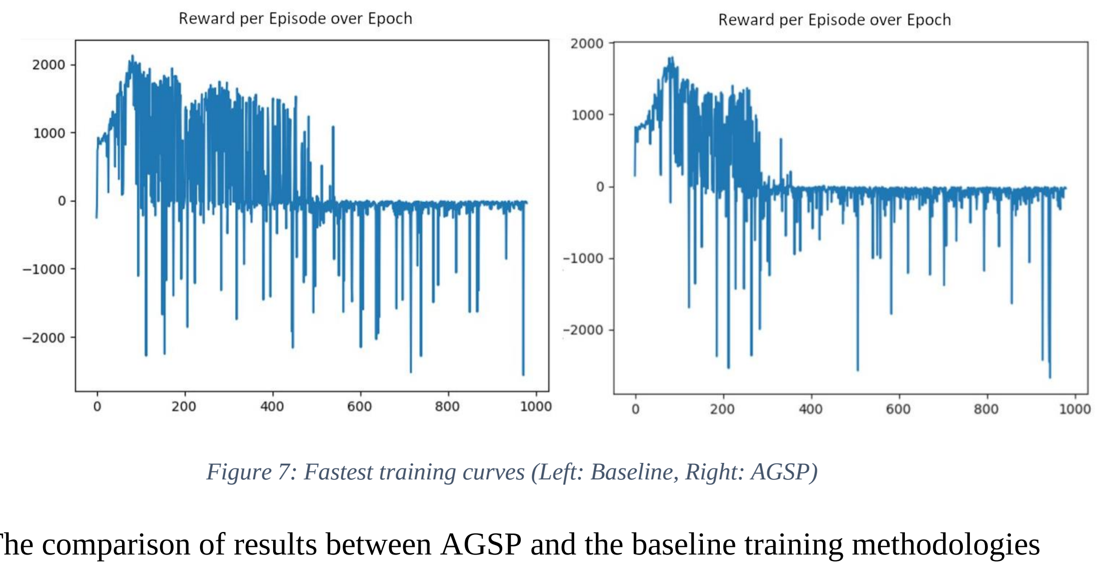

*Figure 7: Fastest training curves (Left: Baseline, Right: AGSP)*

The comparison of results between AGSP and the baseline training methodologies revealed a remarkable similarity in their performance trends. The first test sought to discover if the introduction of AGSP to training would require more experiences to see convergence occur. Initial observations highlight that the number of trials required to achieve convergence to a successful policy remained largely unaffected by AGSP's size or the integration of an additional learning task. Both approaches were able to see successful policies develop after around 80 episodes with the swiftest training epochs shown in Figure 5. The distribution of the most successful episodes was also similar with the baseline taking an average of 350 episodes to hit its peak while the AGSP training averaged 318 episodes to converge to optimal performance as well as having a tighter distribution suggesting that AGSP could potentially promote more consistent learning.

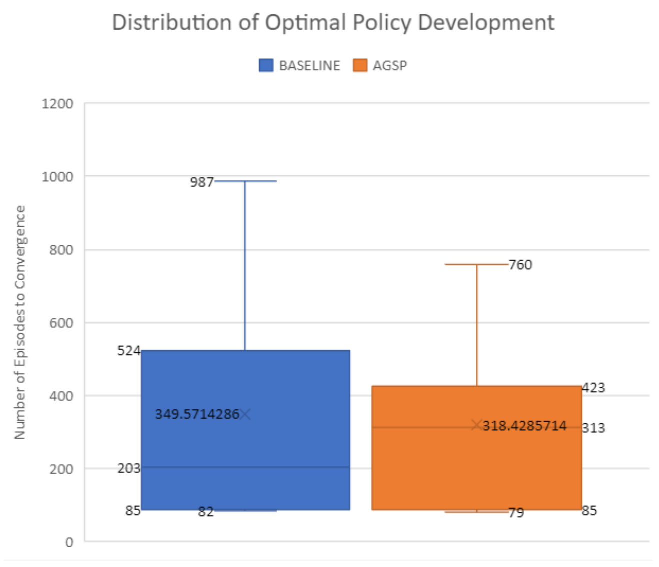

*Figure 8: Distribution of Optimal Policy Development*

However, a pronounced change was observed in the real-time costs incurred by the AGSP methodology. This shift can be attributed to the heightened computational overhead associated with training multiple models and the runtime tasks involved in supplying information to the prediction model. The real time cost increase is one of the most notable changes with AGSP taking longer to run due to the increased computational cost associated with training multiple models as well as the runtime tasks required to provide the information to the prediction model. The true time difference between the methodologies is difficult to measure due to the variety of factors that affected the runtime but the time difference between the methods ranged from negligible up to a 66% increase in time cost in the worst case.

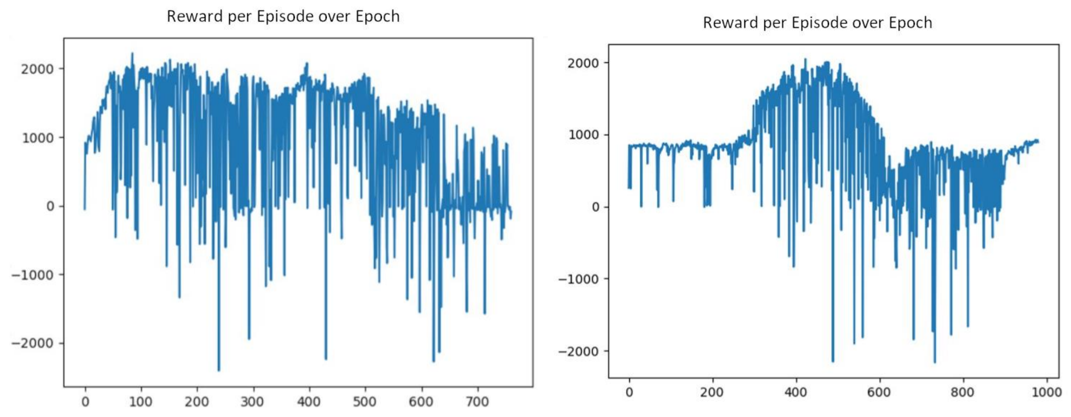

*Figure 9: Training curves for successful policies in four-legged configuration (Left: Baseline, Right: AGSP)*

The subsequent pivotal characteristic under examination was the optimality of the policy achieved through the two methodologies. Focusing initially on the relatively simpler 4-legged locomotion task, it emerged that the unassisted training method consistently demonstrated outcomes with a 7% peak performance (Figure 6) increase compared to the AGSP-enhanced approach. A closer inspection of the nuanced behavioral aspects revealed that this distinction predominantly stemmed from variations in efficiency. While policies trained using both methodologies exhibited comparable walking velocities spanning the range of 1.2 m per timestep to 1.8 m per timestep, a notable divergence surfaced in the efficiency of the gaits produced. AGSP-influenced policies incurred a more pronounced penalty from the control cost component, leading to a diminished overall efficiency in traversing the same distance albeit very slight.

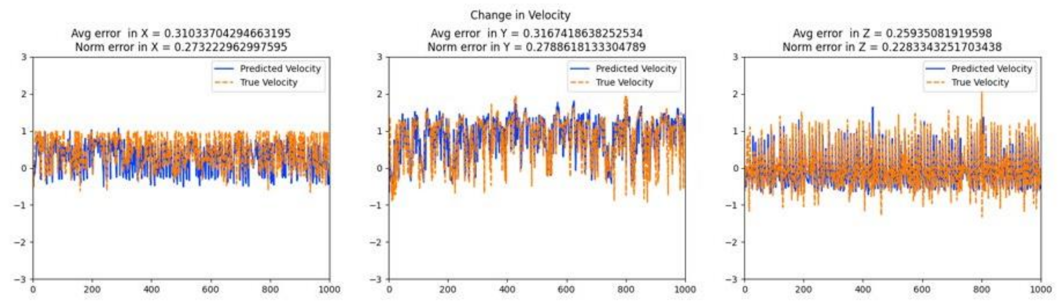

*Figure 10: GSP Output and Error*

Navigating the intricate kinematics of the ant model posed considerable challenges, with its velocity undergoing rapid and unpredictable changes in both direction and magnitude, leading to frequent spikes and an overall noisy trajectory. Ensuring the viability of the Attention-Based Global State Prediction (AGSP) technique hinged upon its ability to accurately predict these abrupt spikes and complex motion patterns within a useful degree of accuracy otherwise it would not have an impact on the outcome of the experiment. In testing the AGSP network consistently demonstrated its aptitude in forecasting velocity changes resulting from the executed actions, maintaining a margin of accuracy within 30%. The shape of the predicted change tightly matches the widely varying shape of the true change in velocity (Figure 7) even with the erratic shape of the true velocity change. AGSP's capacity to effectively capture and predict these intricate dynamics underscores its potential in facilitating a more nuanced and informed policy learning process. It is worth highlighting the interplay between AGSP and the walking policy's performance. Typically, more successful episodes were characterized by higher rewards for both AGSP and the walking policy, reflecting a mutually reinforcing relationship where improved predictions positively influenced policy learning, and vice versa. Since the output of AGSP is not able to match exactly to the true change of velocity at each time step there is the potential for improvement. A hybridized approach that uses a more conventional predictive method while using RL to train the controller could see potentially similar or improved performance with less training required. Additionally, it would be clearer where the error is coming from that cannot be discerned from an RL policy that is opaque to analysis.

To evaluate the consistency of the learned policy we ran trained models for 100 episodes and recorded the percentage of times that the episode failed or was unable to complete the task within reason. The goal of this test was to determine whether the trained model was able to adapt to a variable start in order to evaluate its flexibility and the robustness of the policy to a variety of starting configurations and performance noise. In the 4-legged configuration the performance is similar between the policies trained with and without the assistance of AGSP. The baseline performance saw a failure rate of 6% and the training with AGSP saw a failure rate of 3%. Both outcomes are largely successful highlighting the overall robustness of decentralized control to environmental uncertainty. Additionally, the baseline had several cases of catastrophic failure where the ant ran in the opposite direction leading to the episodes with large negative reward.

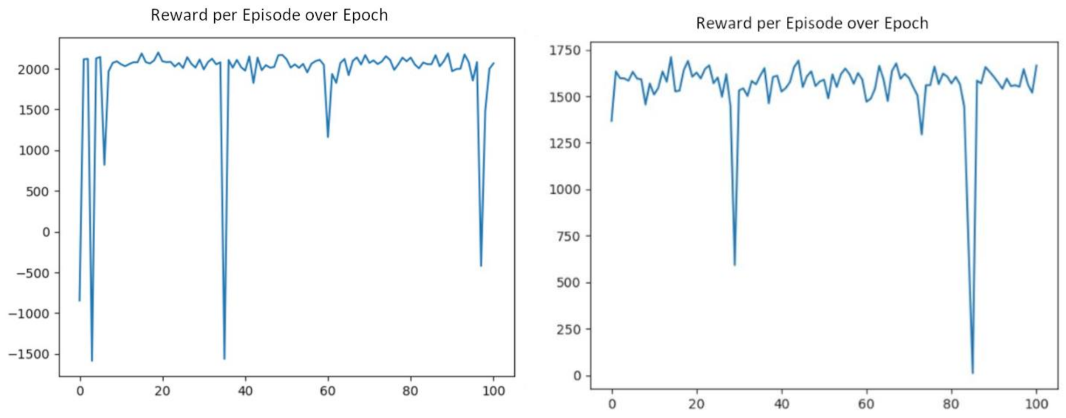

*Figure 11: Trained policy performance for 4-legged configuration (Left: Baseline, Right: AGSP)*

One of the key parameters we wanted to evaluate was the ability of AGSP to provide insight into larger swarm sizes. While the 4-legged configuration provides insight into the general performance and limitations of AGSP the task is not exceedingly difficult or complex enough to necessarily warrant the added overhead of implementation. However, as the scale of the system grows the benefits of implementing AGSP become much more readily apparent. Once again, the training is quite similar across the board, however AGSP does see marginally higher performance on the task.

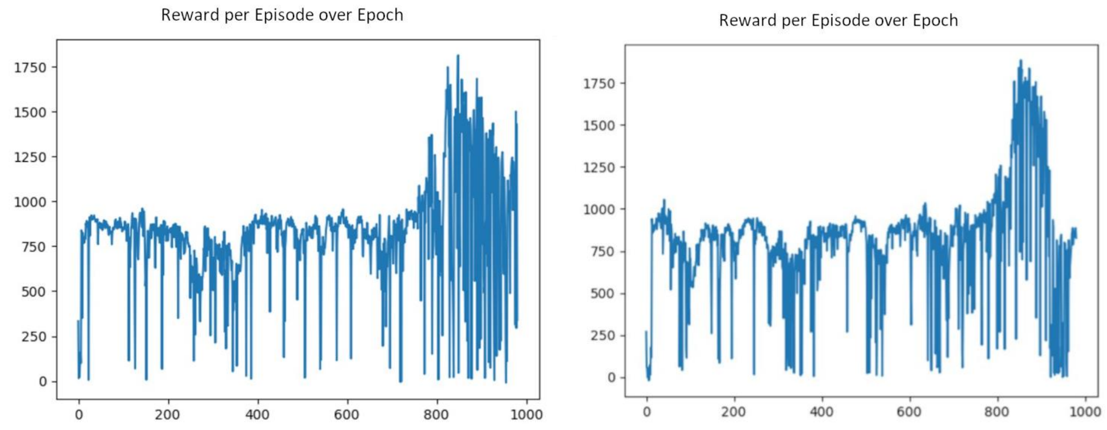

*Figure 12: Training curves for successful policies in eight-legged configuration (Left: Baseline, Right: AGSP)*

When comparing the trained models the performance disparity can be seen more drastically. The baseline performance saw a failure rate of 84% whereas the AGSP assisted training generated a policy that failed only 35% of the time. The task of 8-legged locomotion is not only more complicated in terms of action and observation space, but also due to the possibility of leg collisions and the difficulty of coordinating a more advanced gait. Because of this seeing consistent performance let alone success is quite difficult, but the policy that uses AGSP saw a much greater likelihood of success.

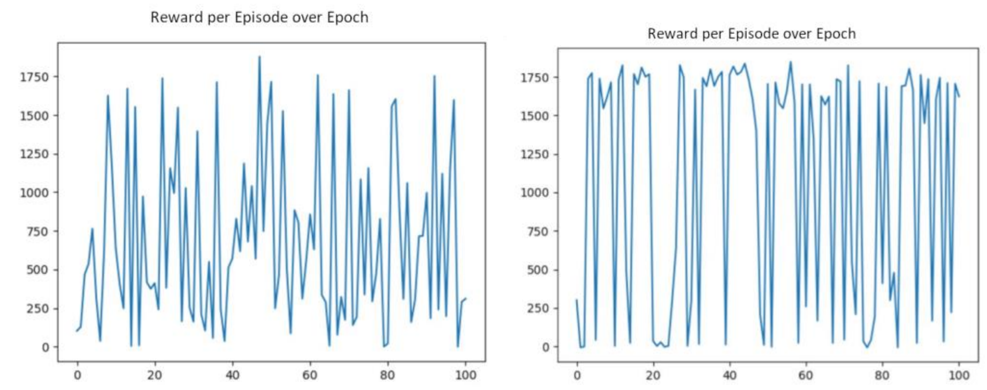

*Figure 13:Trained policy performance for 8-legged configuration*

# Chapter 5

# Conclusion

## 5.1 Summary

This thesis presents a comprehensive exploration of AGSP and its impacts on multi-agent policy learning as swarm size scales. The investigation was driven by five distinct characteristics, each serving as a vital indicator of training success and efficacy. We showed that a single, decentralized model is capable of successfully supporting effective legged locomotion even on an agent with a complex action space.

AGSP's predictive accuracy was a central point of inquiry, revealing its ability to consistently forecast velocity changes resulting from actions closely matching both shape and magnitude in its predictions, even in the face of complex and erratic motion patterns. Furthermore, the evaluation encompassed the consistency of learned policies through rigorous testing of adaptability and performance noise. Notably, policies trained both with and without AGSP demonstrated remarkable robustness in a decentralized control framework.

Expanding the analysis to larger swarm sizes, AGSP's potential came to the fore. While both methodologies exhibited comparable training trends, AGSP demonstrated enhanced performance on tasks involving larger systems showing substantial improvement in consistency indicating a viability for use in training for real world system. This observation underscored the scalability benefits of AGSP, that as system complexities increase AGSP can aid in controlling the increased Degrees of Freedom.

The results shed light on AGSP's potential as a valuable tool for enhancing policy learning in diverse scenarios, underscoring its ability to predict and adapt to complex motion patterns and intricate swarm behaviors. The insights gained contribute to a more nuanced understanding of training methodologies and will help to support future research in the realm of decentralized control and multi-agent systems.

## 5.2 Future Work

The focus of this paper is centered on the development and application of AGSP to legged locomotion, shedding light on its effects on the training process. However, ample opportunities remain for delving deeper into the capabilities of this tool. A further study into the correlation between accurate predictions and the success of the walking policy would indicate the most important elements to predict and how AGSP directly affects the agent. This could also include pretraining AGSP and providing it during the training process of the agent to see if it has a prescriptive effect on the walking policy. The environment itself is subject to variation with similar works making changes to the ant and the system dynamics. A heavier ant would require greater coordination to manage the height of the body or introducing obstacles requires the ability to change direction that is not already present in the task.

Beyond deeper scrutiny and evaluation into the behaviors seen in this work AGSP could be utilized as a communication replacement for agents with local or regional information. If a fixed size network was developed that could complete the task, it would allow for true modularity. When changing from the four-legged to the eight-legged environment the observation space of the agent had to grow to provide enough information to successfully complete the task, but the goal was to develop a model could be trained on a fixed number of observations. If a model that had a set observation size was able to successfully complete the task, then adding additional modules would not require an increase in training time, and models trained on smaller swarms could potentially be applied to more complex systems without any additional training at all.

Lastly, while this study has provided valuable insights into the efficacy of Attention-Based Global State Prediction (AGSP) in the context of multi-agent policy learning a compelling direction for future research lies in conducting a comprehensive comparative analysis of AGSP alongside other prominent model-free kinematic prediction methods. Such a study holds the potential to enrich our understanding of AGSP's unique attributes and its performance relative to established techniques such as the Kalman filter, particle filters, and Gaussian processes.

RL predictive methods and traditional methods such as the Kalman filter present distinct advantages and limitations in predictive modeling. RL methods excel in adapting to intricate and uncertain environments, leveraging trial-and-error learning to improve predictions and decision-making. These approaches are particularly well-suited for scenarios where explicit knowledge of system dynamics is lacking or where non-linear relationships dominate. However, RL methods require substantial data for training and might struggle with exploration strategies and model interpretability. In contrast, traditional methods offer stability, analytical solutions, and efficiency when precise probabilistic models are available. Yet, their reliance on accurate assumptions, notably the presence of a gaussian distribution, can hinder performance in situations where uncertainty or non-linearity prevails. Ultimately, the choice between these approaches hinges on the trade-off between model assumptions, computational efficiency, interpretability, and the application's demand for adaptability to uncertain and complex environments. By comparing AGSP to not only these techniques but other RL predictive methods would indicate whether AGSP was not just a beneficial training tool but a competitive prediction method.

While unveiling AGSP's contributions, this study serves as a steppingstone to a broader landscape of exploration. The proposed future works extend the horizons of AGSP's application, from refining its predictive capacities to enabling seamless modularity and enhancing scalability. As AGSP continues to evolve, it holds the promise of advancing the field of predictive modeling and multi-agent coordination, shaping the future of decentralized control and autonomous systems.

# Bibliography

Ahmadzadeh, H., & Masehian, E. (2015). Modular robotic systems: Methods and algorithms for abstraction, planning, control, and synchronization. *Artificial Intelligence*, *223*, 27–64. https://doi.org/10.1016/j.artint.2015.02.004

Alattas, R. J., Patel, S., & Sobh, T. M. (2019). Evolutionary Modular Robotics: Survey and Analysis. *Journal of Intelligent & Robotic Systems*, *95*(3), 815–828. https://doi.org/10.1007/s10846-018-0902-9

Amri, W. Z. E., Hermes, L., & Schilling, M. (2023). Hierarchical Decentralized Deep Reinforcement Learning Architecture for a Simulated Four-Legged Agent. In G. Nicosia, V. Ojha, E. La Malfa, G. La Malfa, P. Pardalos, G. Di Fatta, G. Giuffrida, & R. Umeton (Eds.), *Machine Learning, Optimization, and Data Science* (pp. 265–280). Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-25891-6_20

Bloom, J., Paliwal, P., Mukherjee, A., & Pinciroli, C. (2023). *Decentralized Multi-Agent Reinforcement Learning with Global State Prediction* (arXiv:2306.12926). arXiv. https://doi.org/10.48550/arXiv.2306.12926

Brunete, A., Ranganath, A., Segovia, S., De Frutos, J. P., Hernando, M., & Gambao, E. (2017). Current trends in reconfigurable modular robots design. *International Journal of Advanced Robotic Systems*, *14*(3), 172988141771045. https://doi.org/10.1177/1729881417710457

Busoniu, L., Babuska, R., & De Schutter, B. (2008). A Comprehensive Survey of Multiagent Reinforcement Learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *38*(2), 156–172. https://doi.org/10.1109/TSMCC.2007.913919

Ding, Q., Chen, J., Yan, W., Yan, K., Kyme, A., & Cheng, S. S. (2022). A High-Performance Modular SMA Actuator With Fast Heating and Active Cooling for Medical Robotics. *IEEE/ASME Transactions on Mechatronics*, *27*(6), 5902–5913. https://doi.org/10.1109/TMECH.2022.3190930

Egorov, V., & Shpilman, A. (2022). *Scalable Multi-Agent Model-Based Reinforcement Learning* (arXiv:2205.15023). arXiv. https://doi.org/10.48550/arXiv.2205.15023

Fitch, R., & Butler, Z. (2008). Million Module March: Scalable Locomotion for Large Self-Reconfiguring Robots. *The International Journal of Robotics Research*, *27*(3–4), 331–343. https://doi.org/10.1177/0278364907085097

Foerster, J. N., Assael, Y. M., de Freitas, N., & Whiteson, S. (2016). *Learning to Communicate with Deep Multi-Agent Reinforcement Learning* (arXiv:1605.06676). arXiv. https://doi.org/10.48550/arXiv.1605.06676

Fujimoto, S., van Hoof, H., & Meger, D. (2018). *Addressing Function Approximation Error in Actor-Critic Methods* (arXiv:1802.09477). arXiv. https://doi.org/10.48550/arXiv.1802.09477

Grabowski, R., Navarro-Serment, L. E., Paredis, C. J. J., & Khosla, P. K. (2000). Heterogeneous Teams of Modular Robots for Mapping and Exploration. *Autonomous Robots*, *8*(3), 293–308. https://doi.org/10.1023/A:1008933826411

Gupta, A., Kumar, V., Lynch, C., Levine, S., & Hausman, K. (2019). *Relay Policy Learning: Solving Long-Horizon Tasks via Imitation and Reinforcement Learning* (arXiv:1910.11956). arXiv. https://doi.org/10.48550/arXiv.1910.11956

Heess, N., TB, D., Sriram, S., Lemmon, J., Merel, J., Wayne, G., Tassa, Y., Erez, T., Wang, Z., Eslami, S. M. A., Riedmiller, M., & Silver, D. (2017). *Emergence of Locomotion Behaviours in Rich Environments* (arXiv:1707.02286). arXiv. https://doi.org/10.48550/arXiv.1707.02286

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

Igl, M., Zintgraf, L., Le, T. A., Wood, F., & Whiteson, S. (2018). *Deep Variational Reinforcement Learning for POMDPs* (arXiv:1806.02426). arXiv. https://doi.org/10.48550/arXiv.1806.02426

Karkus, P., Mirza, M., Guez, A., Jaegle, A., Lillicrap, T., Buesing, L., Heess, N., & Weber, T. (2020). *Beyond Tabula-Rasa: a Modular Reinforcement Learning Approach for Physically Embedded 3D Sokoban* (arXiv:2010.01298). arXiv. https://doi.org/10.48550/arXiv.2010.01298

Kelly, S. D., & Murray, R. M. (1995). Geometric phases and robotic locomotion. *Journal of Robotic Systems*, *12*(6), 417–431. https://doi.org/10.1002/rob.4620120607

Kojcev, R., Etxezarreta, N., Hernández, A., & Mayoral, V. (2018). *Evaluation of Deep Reinforcement Learning Methods for Modular Robots* (arXiv:1802.02395). arXiv. https://doi.org/10.48550/arXiv.1802.02395

Lewis, F. L., & Liu, D. (Eds.). (2012). *Reinforcement Learning and Approximate Dynamic Programming for Feedback Control: Lewis/Reinforcement Learning and Approximate Dynamic Programming for Feedback Control*. John Wiley & Sons, Inc. https://doi.org/10.1002/9781118453988

Li, S., Gupta, J. K., Morales, P., Allen, R., & Kochenderfer, M. J. (2020). *Deep Implicit Coordination Graphs for Multi-agent Reinforcement Learning*. https://www.semanticscholar.org/paper/Deep-Implicit-Coordination-Graphs-for-Multi-agent-Li-Gupta/3598f01215a0a1e451e3b6ae90ae6cae1ba5bb76

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., & Wierstra, D. (2019). *Continuous control with deep reinforcement learning* (arXiv:1509.02971). arXiv. https://doi.org/10.48550/arXiv.1509.02971

Liu, L., Liu, X., Gao, J., Chen, W., & Han, J. (2020). *Understanding the Difficulty of Training Transformers* (arXiv:2004.08249). arXiv. https://doi.org/10.48550/arXiv.2004.08249

Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, P., & Mordatch, I. (2020). *Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments* (arXiv:1706.02275). arXiv. https://doi.org/10.48550/arXiv.1706.02275

Ma, X., Karkus, P., Hsu, D., Lee, W. S., & Ye, N. (2020). *Discriminative Particle Filter Reinforcement Learning for Complex Partial Observations* (arXiv:2002.09884). arXiv. https://doi.org/10.48550/arXiv.2002.09884

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

OpenAI, Andrychowicz, M., Baker, B., Chociej, M., Jozefowicz, R., McGrew, B., Pachocki, J., Petron, A., Plappert, M., Powell, G., Ray, A., Schneider, J., Sidor, S., Tobin, J., Welinder, P., Weng, L., & Zaremba, W. (2019). *Learning Dexterous In-Hand Manipulation* (arXiv:1808.00177). arXiv. https://doi.org/10.48550/arXiv.1808.00177

Peng, B., Rashid, T., de Witt, C. A. S., Kamienny, P.-A., Torr, P. H. S., Böhmer, W., & Whiteson, S. (2021). *FACMAC: Factored Multi-Agent Centralised Policy Gradients* (arXiv:2003.06709). arXiv. https://doi.org/10.48550/arXiv.2003.06709

Qu, G., Lin, Y., Wierman, A., & Li, N. (2020). Scalable Multi-Agent Reinforcement Learning for Networked Systems with Average Reward. *Advances in Neural Information Processing Systems*, *33*, 2074–2086. https://proceedings.neurips.cc/paper/2020/hash/168efc366c449fab9c2843e9b54e2a18-Abstract.html

Riedmiller, M. (2005). Neural Fitted Q Iteration – First Experiences with a Data Efficient Neural Reinforcement Learning Method. In J. Gama, R. Camacho, P. B. Brazdil, A. M. Jorge, & L. Torgo (Eds.), *Machine Learning: ECML 2005* (pp. 317–328). Springer. https://doi.org/10.1007/11564096_32

Rodríguez, A., Parr, R., & Koller, D. (1999). Reinforcement learning using approximate belief states. *Proceedings of the 12th International Conference on Neural Information Processing Systems*, 1036–1042.

Schaff, C., Yunis, D., Chakrabarti, A., & Walter, M. R. (2018). *Jointly Learning to Construct and Control Agents using Deep Reinforcement Learning* (arXiv:1801.01432). arXiv. https://doi.org/10.48550/arXiv.1801.01432

Schaul, T., Horgan, D., Gregor, K., & Silver, D. (2015). Universal Value Function Approximators. *Proceedings of the 32nd International Conference on Machine Learning*, 1312–1320. https://proceedings.mlr.press/v37/schaul15.html

Schilling, M., Konen, K., Ohl, F. W., & Korthals, T. (2020). Decentralized Deep Reinforcement Learning for a Distributed and Adaptive Locomotion Controller of a Hexapod Robot. *2020*

*IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 5335–5342. https://doi.org/10.1109/IROS45743.2020.9341754

Schilling, M., Melnik, A., Ohl, F. W., Ritter, H. J., & Hammer, B. (2021). Decentralized control and local information for robust and adaptive decentralized Deep Reinforcement Learning. *Neural Networks*, *144*, 699–725. https://doi.org/10.1016/j.neunet.2021.09.017

Schulman, J., Moritz, P., Levine, S., Jordan, M., & Abbeel, P. (2018). *High-Dimensional Continuous Control Using Generalized Advantage Estimation* (arXiv:1506.02438). arXiv. https://doi.org/10.48550/arXiv.1506.02438

Singh, A., Chiu, W.-Y., Manoharan, S. H., & Romanov, A. M. (2022). Energy-Efficient Gait Optimization of Snake-Like Modular Robots by Using Multiobjective Reinforcement Learning and a Fuzzy Inference System. *IEEE Access*, *10*, 86624–86635. https://doi.org/10.1109/ACCESS.2022.3195928

Singh, G., Peri, S., Kim, J., Kim, H., & Ahn, S. (2021). *Structured World Belief for Reinforcement Learning in POMDP* (arXiv:2107.08577). arXiv. https://doi.org/10.48550/arXiv.2107.08577

Stulp, F., Isik, M., & Beetz, M. (2006). Implicit coordination in robotic teams using learned prediction models. *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, 1330–1335. https://doi.org/10.1109/ROBOT.2006.1641893

Sukhbaatar, S., Denton, E., Szlam, A., & Fergus, R. (2018). *Learning Goal Embeddings via Self-Play for Hierarchical Reinforcement Learning* (arXiv:1811.09083). arXiv. https://doi.org/10.48550/arXiv.1811.09083

Sukhbaatar, S., Szlam, A., & Fergus, R. (2016). *Learning Multiagent Communication with Backpropagation* (arXiv:1605.07736). arXiv. https://doi.org/10.48550/arXiv.1605.07736

Thor, M., & Manoonpong, P. (2022). Versatile modular neural locomotion control with fast learning. *Nature Machine Intelligence*, *4*(2), 169–179. https://doi.org/10.1038/s42256-022-00444-0

Todorov, E., Erez, T., & Tassa, Y. (2012). MuJoCo: A physics engine for model-based control. *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 5026–5033. https://doi.org/10.1109/IROS.2012.6386109

Varshavskaya, P. (2007). *Distributed reinforcement learning for self-reconfiguring modular robots* [Thesis, Massachusetts Institute of Technology]. https://dspace.mit.edu/handle/1721.1/42058

Varshavskaya, P., Kaelbling, L. P., & Rus, D. (2008). Automated Design of Adaptive Controllers for Modular Robots using Reinforcement Learning. *The International Journal of Robotics Research*, *27*(3–4), 505–526. https://doi.org/10.1177/0278364907084983

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2023). *Attention Is All You Need* (arXiv:1706.03762). arXiv. https://doi.org/10.48550/arXiv.1706.03762

Wang, J., Hu, C., & Zhu, Y. (2021). CPG-Based Hierarchical Locomotion Control for Modular Quadrupedal Robots Using Deep Reinforcement Learning. *IEEE Robotics and Automation Letters*, *6*(4), 7193–7200. https://doi.org/10.1109/LRA.2021.3092647

Whitman, J., Bhirangi, R., Travers, M., & Choset, H. (2020). Modular Robot Design Synthesis with Deep Reinforcement Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, *34*(06), 10418–10425. https://doi.org/10.1609/aaai.v34i06.6611

Whitman, J., Travers, M., & Choset, H. (2021). *Learning Modular Robot Control Policies* (arXiv:2105.10049). arXiv. https://doi.org/10.48550/arXiv.2105.10049

Wu, F., Zilberstein, S., & Jennings, N. (2013, August 3). *Monte-Carlo Expectation Maximization for Decentralized POMDPs*. International Joint Conference on Artificial Intelligence. https://www.semanticscholar.org/paper/Monte-Carlo-Expectation-Maximization-for-POMDPs-Wu-Zilberstein/2e0bc5778eb961cfb48c3cf8deda0bda43425c02

Wu, J., Sun, X., Zeng, A., Song, S., Rusinkiewicz, S., & Funkhouser, T. (2021). Spatial Intention Maps for Multi-Agent Mobile Manipulation. *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 8749–8756. https://doi.org/10.1109/ICRA48506.2021.9561359

Zhang, X., Liu, Y., Mao, H., & Yu, C. (2022). Common belief multi-agent reinforcement learning based on variational recurrent models. *Neurocomputing*, *513*, 341–350. https://doi.org/10.1016/j.neucom.2022.09.144