Deep Learning Approaches for Inferring Collective Macrostates from Individual Observations in Natural and Artificial Multi-Agent Systems Under Realistic Constraints

by

Taeyeong Choi

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved November 2020 by the
Graduate Supervisory Committee:

Theodore P. Pavlic, Chair
Andréa W. Richa
Heni Ben Amor
Yezhou Yang
Jürgen Liebig

ARIZONA STATE UNIVERSITY

December 2020

ABSTRACT

A complex social system, whether artificial or natural, can possess its *macroscopic* properties as a collective, which may change in real time as a result of local behavioral interactions among a number of agents in it. If a reliable indicator is available to abstract the *macrolevel* states, decision makers could use it to take a proactive action, whenever needed, in order for the entire system to avoid unacceptable states or converge to desired ones. In realistic scenarios, however, there can be many challenges in learning a model of dynamic global states from interactions of agents, such as 1) high complexity of the system itself, 2) absence of holistic perception, 3) variability of group size, 4) biased observations on state space, and 5) identification of salient behavioral cues. In this dissertation, I introduce useful applications of *macrostate estimation* in complex multi-agent systems and explore effective deep learning frameworks to address the inherited challenges. First of all, Remote Teammate Localization (ReTLo) is developed in multi-robot teams, in which an individual robot can use its local interactions with a nearby robot as an information channel to estimate the holistic view of the group. Within the problem, I will show (a) learning a model of a *modular* team can generalize to all others to gain the global awareness of the team of variable sizes, and (b) *active* interactions are necessary to diversify training data and speed up the overall learning process. The complexity of the next focal system escalates to a colony of over 50 individual ants undergoing 18-day social stabilization since a chaotic event. I will utilize this natural platform to demonstrate, in contrast to (b), (c) *monotonic* samples only from "before chaos" can be sufficient to model the panicked society, and (d) the model can also be used to discover *salient* behaviors to precisely predict macrostates.

To my family and friends,

who always enjoy my stories

ACKNOWLEDGMENTS

First, I would like to thank my parents, Insoo Choi and Dansuk Yoo, for buying me a desktop computer when I was only 10 years old just because I would never have stopped nagging them without it. I am certain that the good memories with that electric childhood friend has led me to come this far to learn about what it could do further. I would say that this academic journey has been one of the toughest challenges I have ever faced in my life, but it has also been the meaningful moments where I could become much more matured and stronger in every aspect. If I boast about having passed the finish line as the accomplishment I have solely achieved, I will be a liar, because in reality, the past half decade of my life has been shaped by the continuous, encouraging interactions with brilliant and generous people around me. I could not mention every single person here due to the space limit, but I would like to express my deep gratitude below to those who have been nearest to me.

My advisor, Dr. Ted Pavlic, needless to say, has been the most influential person for me to more grow as an independent researcher. I have always enjoyed the chats with him, inspired by his knowledge in science and engineering as well as his ability of discovering a useful perspective from two seemingly independent concepts. Thus, I have tried to learn and apply his way of thinking in my research, and I hope this effort could be found also from my future works. I am also thankful for his gentle communications spending countless hours to listen to my ideas, even when they would sound very silly. Without his patience, even I could not have discovered my potential as a scientist.

I have been fortunate to have good friends with whom I could honestly share my ups and downs in my PhD life. I would never forget the special 2 years in Willowbrook Apartment with my roommate, Keju, where we often had eternal conversations overnight on very random topics in the eyes of international graduate students. I

would really miss the times with my Starbucks friend, Xiushuang (Chris), because we went out to grab some coffee at Starbucks almost everyday to have chats to remind each other that we can keep moving forward for graduation. I appreciate Sehyeok's hard work with physical robots from scratch, and he will remain in my memory as a very fast learner and an amazing robotics engineer. Outside ASU, I also thank Joel Nordtvedt and Nancy Peterson at ISC for being good friends to me and my wife, and for continuing to pray for us.

In fact, Chapter 4 in this dissertation could become possible because Dr. Jürgen Liebig in my committee has provided the video data of *Harpegnathos saltator*. I am particularly grateful to Benjamin Pyenson in his lab, who has made substantial efforts for the video recording. I give thanks to other committee members as well: Dr. Andrea Rica, Dr. Heni Ben Amor, and Dr. Yezhou Yang. They all have been very supportive for me and tried to provide critical and constructive feedback for my research in every committee meeting.

Most importantly, my family have always been the best supporters, even while others were skeptical about what I was trusting. First of all, I would like to thank my mom, Dansuk, for always showing her endless love and care throughout my PhD program, from when I just decided to study abroad and to when I was frustrated with many uncertainties as the graduation was getting nearer. I also know that her sisters, my aunts, have continued to follow every news about me to sincerely share any joys or concerns, and I truly appreciate it. In fact, it is my wife, Ha Eun, who has completely changed my PhD life to be much more bright and hopeful since our wedding in 2018 May. I am deeply thankful for her being right next to me with constant encouragements, and thus, I would also like to play a similar role when she faces any challenge in her life.

As a Christian, I would like to give the deepest gratitude to my Lord, Jesus Christ,

and close these not-yet-sufficient acknowledgements with the Bible verses that have been a powerful guidance for me for the last several years:

> *But I focus on this one thing: Forgetting the past and looking forward to what lies ahead, I press on to reach the end of the race and receive the heavenly prize for which God, through Christ Jesus, is calling us. (Philippians 3:13b-14, NLT)*

I pray that whether my past has been bright or dark, I can always take brave and wise steps to lead the best present and future.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Chapter 1

INTRODUCTION

Since long ago, there have been continuous attempts to consider a complex social system not simply as a set of the membership but as a *monolithic* agent. A very old example is the "Leviathan" published in 1651 (Hobbes, 1904), in which Hobbes represented a society as a huge crowned figure (Fig. 1.1) whose body parts are composed of numerous citizens who have granted a sovereign power to the government as members of a large commonwealth. Ecologists have developed views of social insect colonies as "superorganisms" within which individual insects show behavioral patterns for their colony just as cells behave to maintain an organism; for example, foraging ants can be compared to nutritive cells, egg layers to gametes, and soldiers to immunocytes (Emerson, 1939). Even further, more recent studies have discussed the concept of "collective minds" (Couzin, 2007) where animals in a swarm can virtually construct a collective panel sensor covering the entire team by using their local interactions as an information channel that enables each individual to sense stimuli occurring beyond its actual sensory range.

Such views could then naturally lead a following question: *"Could we define the state of the macro entity itself as the large-scale counterpart of the state in* micro *individuals so as to use them to gain a useful summary of the whole system?"* This approach is more motivated by the notion of "macrostate" in physics, which can provide a better understanding of a system than "microstate" in many cases where, for example, the temperature will be more intuitive information than the location or kinetic energy of every single molecule in the air when controlling the heater in a house.

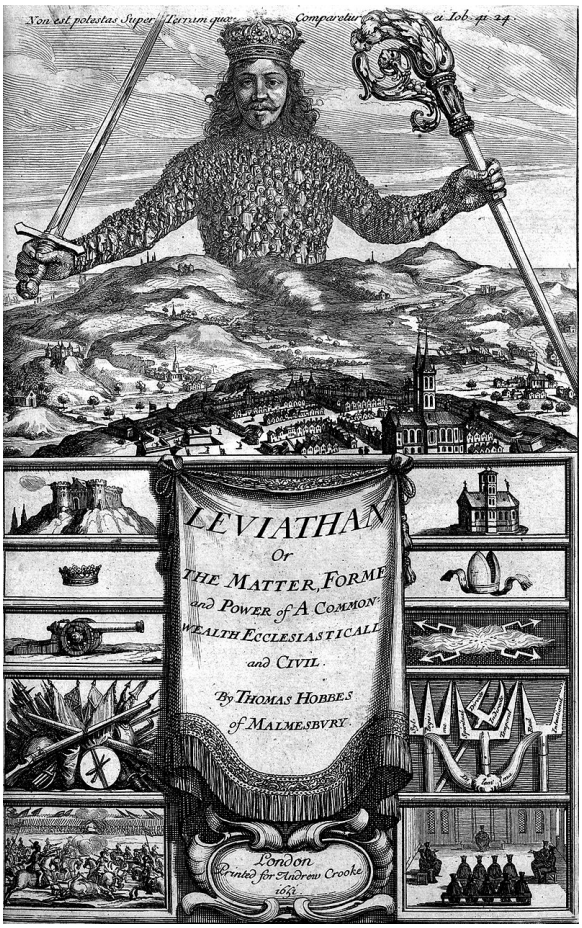


Figure 1.1: Frontispiece of *Leviathan*, reprinted from (Hobbes, 1904)

In this spirit, this dissertation explores applications of *macrostate estimation* on artificial and natural multi-agent systems in which predictive frameworks are required to obtain accurate awareness of useful holistic attributes from the observed local interactions. Within robot teams, for instance, Remote Teammate Localization (ReTLo) is developed for an individual robot to estimate the *global structure* of the team only by utilizing the sequential local observations of a nearby neighbor. Analogous to "collective minds" (Couzin, 2007), the ability allows the localizing robot to expand its sensory range beyond the actual, and it can then wisely choose whether to switch its motion rule to swiftly shift to a particular destination that could lead a better team

formation to succeed in given task. Technically, I will show (a) learning a model of a *modular* team can generalize to all others to gain the global awareness of the team of variable sizes, and (b) *active* interactions are necessary to diversify training data and speed up the overall learning process.

The next platform I will explore is the significantly more sophisticated society in which over 50 ants are undergoing drastic transitions of societal states for 18 days in terms of *social stability*. I will utilize this natural colony to demonstrate, in contrast to (b), (c) *monotonic* samples only from "stable" can be sufficient to model the "unstable" society, and (d) the model can also be used to discover *salient* behaviors to precisely predict macrostates.

Before we deeply investigate the concrete applications, I will first describe the concept of "macrostate" and the "estimation problem" that this dissertation handles. In addition, anticipated challenges that can be accompanied in realistic scenarios will be discussed, and then I will provide a brief overview of this dissertation.

1.1 Concept of Macrostate: *From "micro" to "macro"*

According to Poole and Mackworth (2010), a state of an agent must be an instance that contains all the properties necessary to predict the effects of an action taken and assess if the agent has reached a goal state. We slightly extend this general condition to apply to a larger macro entity within which a number of micro-agents can actively interact with the environments – i.e., a *macrostate* refers to an instance of properties necessary to predict the effects of a *joint* action taken and assess if the *macro agent* has reached a goal state.

A macrostate is not necessarily equal to a joint state widely used in problems of multi-agent planning, where by definition, a set of states from micro-individuals are simply concatenated resulting in a representation of a higher dimension, because it

can be more *general* to include any properties to holistically characterize the system. A similar example is the macrostate in physics, where, for example, the volume of a gas cannot be obtained by a simple concatenation of any microlevel states, such as locations of molecules, although there could be a functional mapping.

Furthermore, a macrostate can be obtained from different microstates; in ReTLo task, for instance, the team-level quality to estimate is defined by the positions of all individual robots, and hence, the robots can actually possess different combinations of orientations to provide the identical macro formation. Similarly, for an ant colony, the same level of social stability could be maintained even though a particular ant has different social ranks at the times of measurement.

1.2 Macrostate Estimation

As stated above, a complex multi-agent system can be deemed as a single large agent, no matter how many individual members inhabit, experiencing a trajectory of state transitions over time as depicted below:

$$\cdots \rightarrow M_{t-1} \rightarrow M_t \rightarrow M_{t+1} \rightarrow \cdots$$

where M_t is the macrostate at time instant t, and $\rightarrow$ simply denotes the temporal order of the observed state evolution. In fact, the temporal transitions are primarily driven by local interactions that micro-agents make – $L_t \triangleq \{\ell_{1@t}, ..., \ell_{N@t}\}$ where $\ell_{i@t}$ is the encoded interaction of agent i at time t leading:

$$\cdots \xrightarrow{L_{t-1}} M_{t-1} \xrightarrow{L_t} M_t \xrightarrow{L_{t+1}} M_{t+1} \xrightarrow{L_{t+2}} \cdots$$

where $a \xrightarrow{b} c$ abstracts underlying stochasticity in the transition from a to c after b executed. Finally, the estimation problem can now be defined as follows:

Definition. *Macrostate Estimation* is to reconstruct a macrostate $M_{t+\Delta}$ evolved from

a past state M_t using the temporal observations $O_{t:\Delta'} \subseteq L_{t+1} \cup L_{t+2} \cup \cdots \cup L_{\Delta'}$ where $0 < \Delta \leq \Delta'$.

Here, observations $O_{t:t+\Delta'}$ are involved to consider practical scenarios in which only a subset of local interactions are observed by the estimator f for a certain period of time possibly due to its limited sensory capability or the occlusion between agents. Also, note $\Delta \leq \Delta'$ implying that the observations $O_{t:\Delta'}$ could be used to predict a historical state $M_{t+\Delta}$. In this dissertation, therefore, we will apply this general framing to useful applications of macrostate estimation, whether the focal system is artificial or natural, and our goal is to build the best estimator f^*, which can compute the estimate $\hat{M}_{t+\Delta}$ that causes the minimum average error from the actual macrostate $M_{t+\Delta}$.

1.3 Challenges

As macrostate estimation is conducted on a sophisticated social system, there could be challenges that must be addressed to obtain a reliable estimator especially when the environment where the system is deployed imposes realistic constraints. I suggest six potential issues below and for each, will describe promising solutions that could have connections with applications in the following chapters.

***Challenge 1 (C1)*: High complexity of system itself**

As more agents are involved, a multi-agent system generally becomes more *complex* since there are more possible combinations of "microscale" interactive actions and states from the membership in order to predict the "macroscopic" outcome. In this respect, all the systems we will explore in this dissertation can be said to be complex in that robots or insects make frequent interactions with the environments leading diverse collective properties to emerge that a single micro-member alone or a less

complex group could not promote.

To model such complexity, I mainly investigate deep neural network models, in which the complexity can be determined by the number of parameters to be optimized, and sufficient *depth* could allow the hierarchical structure to learn sophisticated relationship from "micro" to "macro" layer by layer. In addition, *modularization* of the system could also be effective to model the dynamics in a "less complex" subgroup and generalize to all other modules to finally obtain a holistic representation. We will more deeply explore the technique for ReTLo in Chapter 2.

***Challenge 2 (C2)*: Absence of holistic perception**

Observations of some individuals may not be available for the estimator in realistic contexts where sensory and communicational resources are limited to only a subset at a time. For example, a robot in a robotic team could only observe others located within its sensory range, and robot-to-robot communication could also be disabled or discouraged to save communication resources or hide from adversarial agents. In this case, the motional interactions with the neighbors could be used as the only medium to gain the global awareness beyond the sensible region. ReTLo problems in Chapter 2 will focus on this issue to suggest learning motional dependency of teammates to complement the restricted capability of perception and communication for better cooperative coordination.

***Challenge 3 (C3)*: Variability of group size**

An estimator fitted to a particular system may fail to generalize to another if the number of involved agents changes causing more or less frequent local interactions. In other words, re-training could be needed to model the dynamics of the system whenever the membership changes. To tackle this challenge, *modularization* could be

a potential solution by which the estimator is designed to perform within a subgroup of a meaningful size so that repeating applications of it across other groups could produce a synthesized inference about the entire team. Although a system becomes larger or smaller, consequently, the modular estimations can simply adjust the number of executions without needing failing the model. Chapter 2 will show how these techniques can be incorporated into multi-robot teams.

***Challenge 4 (C4)*: Biased observations on state space**

Learning an effective model of macrolevel states requires sufficient exposure to each unique state to collect representative data. Some states, however, might not be observed without any special stimulus to lead a particular condition. In robotic applications (Chapter 3), I will introduce an *active* motion rule for an individual robot to better promote the team to lead more diverse properties and minimize the bias in observations for learning. In Chapter 4, an alternative approach on ant colonies will allow for *monotonic* training datasets to represent only the "normal" state while the model will also be able to classify the "abnormal" state. Such a challenge can easily be found in biological or human societies, in which the datasets of normal behaviors, e.g.) walking pedestrians, are often much cheaper to gain than those of the abnormal, e.g.) rioting.

***Challenge 5* (C5): Identification of salient observations**

A complex social system continuously involves numerous microlevel interactions of various types, which thus should be informative cues of emerging macrostates. It could, however, be highly cryptic which local behaviors trigger a particular macrostate especially if a high degree of stochasticity is present. Deep-learning algorithms could be advantageous in this case since they have been utilized when key features cannot

be specified in advance, so the model must automatically discover them from the training data. Even further, in Chapter 4, we will see the potential application of Grad-CAM (Selvaraju *et al.*, 2017) on ant colonies to locate specific ants that are engaged in *salient* micro-interactions to lead macrostates of the society.

1.4 Dissertation Outline

The following is a brief overview of this dissertation to deal with the subjects mentioned above:

- **Chapter 2** — I start with the ReTLo problem in multi-robot teams, in which an individual robot uses microscopic behavioral observations from its neighbor to infer the macrolevel formation states. A "modularized" prediction scheme is suggested to effectively overcome **C1∼C3** on differential drive robots in simulations. Not only the inference is validated to be accurate but also a use case in caging scenario is showcased. To obtain more reliable estimation, furthermore, sequential observations are considered as input on physical robotic platforms demonstrating significant improvement over the case where only transient behaviors are used.

- **Chapter 3** — This chapter focuses on the risk of **C4** that the robotic teams could accompany while learning ReTLo. Unlike the previous approach, the robotic learner here becomes "active" in motion selection to acquire non-biased experiences of the team dynamics during data collection. Additional models are introduced predicting the novelty levels of possible actions to only take the optimal among them.

- **Chapter 4** — I deal with a natural system, an ant colony, as a testbed with **C4** to detect the "unstable" social state although only "stable" behaviors are ob-

servable while learning a macrostate estimator. A generative model is developed to create fake "ideally stable" behaviors on purpose with which a binary classifier is trained to successfully perform the macrostate estimation. Moreover, "salient" behavioral features are located for different macrostates, to bridge the knowledge gap between microlevel interactions and macro-scale states (**C5**).

- **Chapter 5** — I will conclude the dissertation summarizing contributions and potential directions of future work.

Chapter 2

REMOTE TEAMMATE LOCALIZATION [1]

We begin with *macrostate estimation* in robotic multi-agent systems in which the Remote Teammate Localization (ReTLo) problem is introduced and addressed. In particular, "formational properties" of the complex set of agents are inferred for individual robots to use the situational information to better coordinate the trajectories of the team eventually. To respect realistic assumptions, the problem is designed to accompany some of the challenges listed in Section 1.3 – i.e.,) the macroscale structures depend on continuous interactions of micro-agents (**C1**), every robot can only perceive local events (**C2**), and the learned model must be scalable to varying sizes of the robot team (**C3**). Using local interactions to solve the ReTLo problem could be interpreted in the view of behavioral ecologists like Nikolaas Tinbergen (Bradbury and Vehrencamp, 2011) that "implicit cues" in behaviors could be converted to "explicit signals". Couzin has also made a relevant point in his article "collective minds" (Couzin, 2007) that close behavioral coupling that is local but continuous across an animal swarm can virtually create a "self-organizing array of sensors" by which an individual can gain useful information beyond its actual sensory power. Through this chapter, we will examine the realization of such biological insights on both simulated and physical robotic platforms and also discuss the achievable benefits

[1] This chapter is based upon (Choi *et al.*, 2017) and (Choi *et al.*, 2020).

and limitations.

2.1 Introduction: *Sensing beyond sensors*

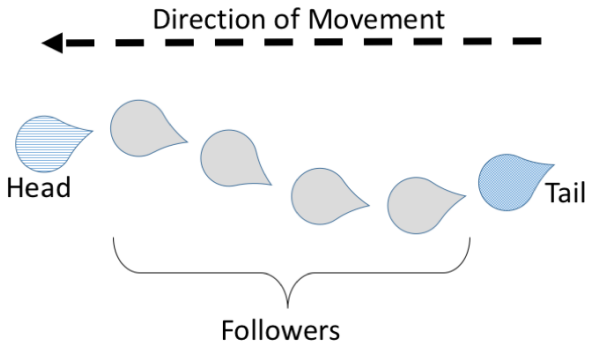
In networked multi-robot teams, coordinated *macro*-behaviors typically emerge from *micro*-interactions among adjacent robots with limited sensor range, simple motion rules, and no persistent connection to any centralized command and control authority. In principle, such a decentralized approach naturally scales well to real-world problems where a large number of robots must be deployed over relatively large areas where constant global communication is impractical. Significant effort has been focused on the design of decentralized control algorithms for multi-robot systems (Wang and Kumar, 2002; Correll and Martinoli, 2004; Odhner and Asada, 2010; Napp and Klavins, 2011; Brambilla *et al.*, 2013; Rubenstein *et al.*, 2014; Wilson *et al.*, 2014; Derakhshandeh *et al.*, 2014; Yang *et al.*, 2015; Valentini *et al.*, 2015; Valentini and Hamann, 2015; Elamvazhuthi and Berman, 2016; Valentini *et al.*, 2016). However, approaches that make use of high levels of communication among agents present scalability challenges, and approaches that eschew communication often achieve less than ideal coordination as a consequence.

If a single robot could, with little explicit communication, gain awareness of the status of other very remote agents in the team or awareness of some macroscopic state of the group, then a new class of highly scalable coordinated behaviors would be possible. Robots in one location could perform behaviors that complement the actions of robots in another location in a way that is currently only possible using levels of signalling that are particularly burdensome in large-scale groups. It is now possible to shift from explicit to implicit communication, from what behavioral ecologists call "signals" to "cues" (Bradbury and Vehrencamp, 2011), because of the recent increase in computational power available on modern small robotic platforms. Whereas the

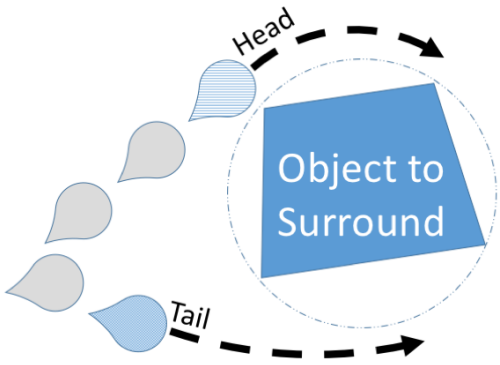
platforms of the past compensated for little local computational power with communication to coordinate with other agents, robotic platforms of the future can rely on high levels of local computational power to reduce the need for communication that may be energetically costly or physically infeasible.

Toward this end, we propose machine-learning methods to solve the ReTLo problem where a robot (*Tail*) at one end of a line formation of a multi-robot team is to predict positions of all other teammates only using local observations about a single nearby teammate. Because each robot has a limited sensor radius and a relatively simple motion rule that depends on the position of its nearest neighbors, the *Tail* has to be able to learn the regularity of the observed motions of its neighbor to finally infer the poses of all other robots. We introduce a repetitive prediction scheme to use predictions about nearer teammates to make predictions on farther ones until the prediction reached the *Head* robot at the other end in line formation. In a multi-robot simulation, we showed the feasibility of using the method in an example caging scenario in which the *Tail* could recognize the early stages of a caging action of *Head* and promote a proactive maneuver to better assist in coordinating the team to quickly enclose an encountered object in the environment.

To demonstrate our approaches, we consider a simple scenario where a heterogeneous multi-agent system like the one in Fig. 2.1a maintains a chain formation and consists of three types of nonholonomic mobile robots: *Head*, *Follower*, and *Tail*. *Head* is a robot at one end of the chain formation, *Tail* is another at the distal end, and between them is a string of *Follower* robots. Our goal is to train artificial neural networks (ANNs) (Schmidhuber, 2015) on *Tail* to learn how to utilize locally observable information – the position of its nearest neighbor – to estimate the position of every robot in the system even including the *Head* when it is outside of the sensor range of the *Tail*. The major strength of our approach is that training of the ANNs on



(a) Multi-robot chain



(b) Caging scenario

Figure 2.1: Motivating examples. Robots move in a chain formation until encountering an object to be encircled. In (a), heterogeneous robots move in chain formation from right to left. In (b), the *Tail* robot breaks out of chain formation to more rapidly encircle an object encountered first by *Head*.

Tail requires only a 3-robot chain, and the resulting ANN can be applied by *Tail* to fix the positions of robots in longer chains simply by applying the ANN repeatedly in a recursive fashion. We first validate our models in a simulation of robots with realistic assumptions – the presence of noise and little-or-no communication availability – and also show that the approach can facilitate coordinated behaviors like the one in Fig. 2.1b. A more developed model with Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) is then proposed to demonstrate its high reliability on a physical, commercial robotic platform, *ThymioII* (Riedo *et al.*, 2013), with the quantitative and qualitative results.

2.1.1 Chapter Highlights

A brief overview of the subjects in this chapter is as follows:

- **Section 2.3** – We will formally build a multi-robotic system under realistic constraints in which individual mobile robots can perceive only their nearest neighbors as the team is driven by a *Leader* robot in the chain formation.
- **Section 2.5.1** – The core technique of *modular* macrostate estimation is introduced. The estimation process is first explained in a short module of 3 robots first, and then the example is scaled up to a larger team. The same modularization method is applied in both the simulated and physical platforms only with a difference in designs of the main estimator applied.
- **Section 2.5.2** – Specific settings on experiments are described including the *Robotarium* simulator Pickem *et al.* (2016) and the hyperparameters for the neural network models.
- **Section 2.6.1** – Improved estimator with LSTM layers is introduced to effectively encode a longer period of observations than the previous feed-forward

networks, to achieve more accurate predictions on real robots.

- **Section 2.6.2** – We will learn the experimental environments on *ThymioII* (Riedo *et al.*, 2013) used to gather data samples for training and test. In particular, practical strategies are explained to generate sufficiently challenging trajectories for fair evaluations.

2.2 Related Work

In this section, we discuss results from the multi-robot systems literature similar to ours and elaborate on the distinct contribution of our work.

Cooperative localization and tracking

Although the ReTLo problem is superficially similar to other known cooperative localization and tracking problems (Grocholsky *et al.*, 2004; Luft *et al.*, 2016; Franchi *et al.*, 2010; Cheng and Xie, 2014; De Silva *et al.*, 2015), such as cooperative SLAM, it is distinct from general robotic localization. In ReTLo, the robot does not execute predictions on its own location but on its teammates using accessible information. Moreover, in contrast with cooperative localization approaches, robots in ReTLo are assumed to be communication free and thus not allowed to communicate with other members during the prediction of positions. Hence, in lieu of direct signaling, an underlying assumption of the ReTLo problem is that the robot behaviors are correlated with the state of the environment around them and thus contain cues about the state of their neighbors. In this sense, state observers in a networked robotic system are more similar to ReTLo than robotic localization (Xiao-Yuan *et al.*, 2010; Antonelli *et al.*, 2012), but ReTLo does not depend upon knowledge of the structure of the underlying robotic controllers and does not require that robots move according to simplistic, analytically tractable dynamical models. Furthermore, we emphasize

ReTLo solutions focused on scaling from training with small teams to implementation in potentially much larger and possibly variably sized teams.

Group state recognition

The ReTLo problem aims to infer the positional state of an entire multi-robot team using only locally obtainable information in the aspect of a robot member. This is because the knowledge about global configuration could help make a better decision for the sake of whole team. In this spirit, Brown and Goodrich (2014) as well as Berger *et al.* (2016) show that local interactions of robots within a swarm can be used to classify swarm-level macroscopic structures, such as particular swarm-level shapes like *flock* and *torus*. In contrast, our work is to estimate the pose of robots themselves, which are the microscopic elements of the multi-robot team. Consequently, we make inferences on a space with far more degrees of freedom and require a more powerful regression model.

Behavioral cue interpretation

In the ReTLo problem, the pose of remote robots is to be inferred from the motions of nearby robots performing otherwise nominal behaviors. This approach allows information to flow around a multi-robot team without traditional communication modalities for explicit signalling, such as radio communication. Motivated by similar constraints on reducing the use of these modalities, Novitzky *et al.* (2012) and Das *et al.* (2016) showed how robots performing a special behavior, similar to a "waggle dance" of honeybees (Von Frisch, 1967), could convey information visually or mechanically to remotely observing robots. Their approaches are different from ours in that we do not require robots to deviate from their normal behaviors for the purposes of explicit communication; we infer positions only from the latent information in nominal robot

behavior and interactions.

Robot dynamics learning

Byravan and Fox (2017) proposed a deep learning approach to predict the next visual frame given both a current visual frame as well as knowledge of a force acting on an object within the frame. One of the motivations for this work was to understand the dynamics of robotic arms and the relationship with control commands possibly executed. In the ReTLo problem described, the *Tail* robot may have accumulated a global shape of robot team over time, and it has to be able to predict the future formation as a new observation on its neighbor is provided, which could be viewed as gaining knowledge of an applied force on the team. Such a similarity inspired the architecture of our neural network model, but Byranvan and Fox focused on learning motions of rigid objects whereas a chain of robots in our work can present much flexibility in team shape. In addition, the positional information about the nearest neighbor is only loosely analogous to the perfect knowledge of force used in the frame-prediction example. Consequently, our approach is a significant deviation from the one proposed by Byravan and Fox (2017).

Robot teams in chain formation

In our running example of a moving chain formation, the formation can be driven by two heterogeneous leaders, *Head* and *Tail*. A similarly structured robotic system was designed by Elamvazhuthi and Berman (2016) consisting of a long string of holonomic point robots that are terminated by two *leader* agents. While the interior robots in the chain perform simple, wave-like motion rules based on position information from neighbors only, the two leader robots have pre-programmed trajectories that are guaranteed to drive the interior robots to desired locations along a formation.

In contrast, our goal is to infer remote properties of the multi-robot team based on local information even when the individual-robot motion models are too complex to be analyzed mathematically. Thus, our methods generalize to more realistic robotic motion rules – such as nonholonomic robots limited to two-dimensional planar motion with collision avoidance.

Caging behaviors in multi-robot systems

Our motivating application example, caging, is a popular scenario in the robotics literature. Wilson *et al.* (2014) developed a stochastic robotics method for boundary coverage meant to mimic the collective transport behavior of ants, and Derakhshandeh *et al.* (2014) developed an amoeba-inspired distributed algorithm for coating objects on a specially defined grid. Each of these applications makes use of identically controlled agents that come to equilibrium in some desired formation, often with the assumption of a simplistic motion model. In contrast, we focus on heterogeneous multi-agent systems where some agents must gain global situational awareness in order to make decisions about whether and when to switch from certain behavioral modes to certain other modes. In principle, our method could be used in combination with these other techniques so that agents can determine when to switch to other behaviors after the desired configuration has been reached.

2.3 System Design [2]

We consider a multi-agent system consisting of n nonholonomic mobile robots that move within a chain formation in the plane of $\mathbb{R}^2$. At all times, each one has a pose denoted by a three dimensional vector (x, y, θ) where x and y specify its position and θ its orientation. Every robot is controlled by a decentralized controller and can only

[2](Choi *et al.*, 2017)

detect the distance and angle to another robot or other object within a limited sensor radius. Due to this limitation, each robot cannot localize in the global coordinate system but can use their own frame of reference to represent observed objects.

Each robot type from Fig. 2.1a has a distinct motion rule.

- The *Head* moves independently in the given space.
- The $n-2$ *Follower* robots continuously update their position to maintain a constant distance between their two nearest neighbors.
- The *Tail* either follows its single neighbor at a constant distance or moves independently.

So as the *Head* moves, it pulls a nominally rigid chain of robots behind it, but *Tail* can switch to moving independently of the others, causing the chain to be pulled at both ends.

Formally, we define the set of robots $\mathcal{R} \triangleq \{1, 2, \ldots, n\}$ where robot n is the *Head*, robot 1 is the *Tail*, and robot $i \in \{2, \ldots, n-1\}$ is a *Follower*, interchangeably denoted as *Follower* $(i-1)$ where $(i-1)$ indicates the number of robots ahead of the *Tail* (i.e., *Follower* 1 is closest to the *Tail*). Every robot $i \in \mathcal{R}$ is assumed to abide by nonholonomic unicycle kinematics of the form

$$\begin{cases} \dot{x}_i = v_i \sin(\theta_i) \\ \dot{y}_i = v_i \cos(\theta_i) \\ \dot{\theta}_i = \omega_i \end{cases}$$

where vector $\vec{p}_i \triangleq (x_i, y_i)$ is the robot's Cartesian position, θ_i is the robot's orientation, v_i is the linear velocity of the robot, and ω_i is the angular velocity of the robot. The *pose* $\vec{s}_i \triangleq (x_i, y_i, \theta_i)$ of a robot $i \in \mathcal{R}$ is a vector containing its position in the plane and its heading orientation. Thus, by adjusting ω_i and v_i, the nonholonomic

robot can be driven like a unicycle over the plane. We use this model for simplicity. However, the data-oriented machine learning approaches described in Section 2.5.1 and Section 2.6.1 should be applicable to a wide range of other kinematic and dynamic models as well.

To implement the chain-following formation shown in Fig. 2.1a, we use fully actuated, holonomic kinematics in the Cartesian plane to generate *reference trajectories* for ω_i and v_i to track within the more realistic nonholonomic unicycle kinematics. As described above, the underlying robots attempting to track these reference behaviors will do so with some error because they are nonholonomic. So the actual behavioral rules are a composition of simplistic holonomic kinematics with the more realistic nonholonomic kinematics of the robots. The particular coordinate transformation and nonholonomic constraints are implemented under conventional control theory for differential wheeled robots, and so we only present the fully actuated, target motion rules here. In particular:

- The *Head* robot approximates the motion rule:

$$\dot{\vec{p}}_n = \vec{T}_h - \vec{p}_n$$

  where $\vec{T}_h$ is some target Cartesian position in the space. Thus, the *Head* robot is always moving toward some target point set elsewhere in its control hierarchy.

- Each *Follower* robot $i \in \{2, \ldots, n-1\}$ approximates:

$$\dot{\vec{p}}_i = k_f((\|\vec{p}_i - \vec{p}_{i-1}\| - d)(\vec{p}_{i-1} - \vec{p}_i) + (\|\vec{p}_i - \vec{p}_{i+1}\| - d)(\vec{p}_{i+1} - \vec{p}_i))$$

  where k_f is a scalar that scales the difference between the desired spacing around robot i to a desired velocity that should, if geometrically possible, return the *Follower* to a distance of d away from its two neighbors.

- The *Follower* normally behaves like a *Follower* with one neighbor, as in:

$$\dot{\vec{p}}_1 = k_t((\|\vec{p}_1 - \vec{p}_2\| - d)(\vec{p}_2 - \vec{p}_1)).$$

  However, it could switch to an independent motion rule when it has noticed that *Head* has detected some object to encircle particularly to achieve caging missions attempted in Section 2.5.4.

Thus, the challenge is how the *Tail* can use regularities in the patterns of interactions that emerge from such simple motion rules to predict the positions of all teammates, including the farthest *Head*, by using only accessible knowledge such as the position of robot 2, the *Follower* closest to *Tail*.

2.4 ReTLo Problem [3]

The ReTLo is to localize all teammates from the view of *Tail* only using locally observable information of the position of *Follower* 1 at every time step. Here, we introduce a general version of ReTLo with a practical assumption that until a specific time instant τ, all the information about positions and orientations of all robots have been shared reliably with the *Tail* robot, possibly via global communication, but continued information sharing is unavailable after time τ. Consequently, *Tail* must use this localization technique to extrapolate from the previously known reliable positions. Based on this setting, we will actually deal with two different scenarios after time τ where in Section 2.5.1 the *Tail* relies only on most recent observations at each time instant while in Section 2.6.1, all the prior knowledge until τ are also utilized for future estimations.

Formally, at the time instant τ, the following set of poses of all teammates is

[3](Choi *et al.*, 2020)

available:

$$\{\vec{p}_{r@t}, \theta_{r@t}\} \tag{2.1}$$

where $\vec{p}_{r@t} \triangleq (x_{r@t}, y_{r@t})$ is the position of robot r at time t and $\theta_{r@t}$ is the orientation of robot r at time t, with $r \in \{T, F_1, F_2, ..., F_{n-2}, H\}$ and $t \leq \tau$. The ReTLo problem is thus, at each time $t > \tau$, to observe the position of F_1, $\vec{p}_{F_{1@t}}$, and use all available information to predict the pose set in Equation 2.1 for time t' where $\tau < t' \leq t$. In other words, the ReTLo problem is a specific example of *macrostate estimation* (c.f. Section 1.2) in which macrostate $M_{t'} \triangleq \bigcup_{i=1}^{N} \{p_{i@t'}, \theta_{i@t'}\}$ is estimated given the previous state M_τ and the temporal observations $O_{\tau:t} \triangleq \{p_{2@\tau}, ..., p_{2@t}\}$.

2.5 On Computer Simulations [4]

We first investigate the plausibility of ReTLo on computer-simulated robots before real robotic models are implemented. Specifically, a machine learning pipeline which involves relatively shallow feed-forward neural networks is proposed here to use recent positional observations from the nearest follower at each time to predict the pose information of all teammates, although we will see later in Section 2.6 a more advanced model to ensure higher performance on actual robots.

2.5.1 Methodology

Our goal is to design an essentially communication-free algorithm that enables *Tail* to estimate the position of each distant robot $r \in \{F_2, F_3, \ldots, H\}$ in the team using only locally sensed information about robot 2, *Follower* 1 immediately ahead of it. Furthermore, the same algorithm should be able to be used for a wide range of team sizes. The scalability of our approach comes from our use of modular artificial neural networks that are applied recursively as opposed to a monolithic ANN tailored

[4] (Choi *et al.*, 2017)

for a particular team size.

2.5.1.1 Learning from Instant Behaviors

Toward accomplishing our goal, we first consider the simple 3-robot case ($n = 3$) where robot 3 is the *Head*. On *Tail*, we use backpropagation Bishop (2006) to train a set of ANNs, $\{ANN_x, ANN_y, ANN_\theta\}$, as regression models that can predict the Cartesian coordinates of robot 3 as well as the orientation of robot 2 (*Follower*) using only the sensed coordinates of robot 2. They are trained with pose data collected at discrete time instants during interactions among the three robots in simulations where robot 3 takes arbitrary motions.

Each ANN is a multi-layer perceptron employing three layers – one input layer, one hidden layer, and one output layer. The hidden layer utilizes 12 nodes, each of which performs computations that take values from all nodes of the input. Each hidden node uses the logistic, sigmoidal activation function whose output ranges continuously from 0 to 1. For regression purposes, one node set in the output layer has a linear activation function taking outputs of all hidden nodes as input. The major difference in the structure of the three ANNs is in their input layer. The ANN_x, which is used to predict $\vec{x}_{H@t}$, takes six inputs consisting of:

- $\vec{s}_{F@t}$, the pose of robot 2
- $\vec{p}_{F@t+1}$, the relative coordinates of robot 2 at time $t + 1$
- b, a scalar bias

where all positions noted here are assumed to be expressed in the reference frame of the *Tail* robot because, in practice, the *Tail* has to understand positions of others by projection onto the local coordinate system centered at itself. Though a similar conversion may be considered for orientation, we use absolute direction in this work

(i.e., assume that all robots agree on compass directions) for simplicity. In training, the ANN_y that estimates $\vec{y}_{H@t}$ uses these same six inputs plus $\vec{x}_{H@t}$. Similarly, the ANN_θ for $\vec{\theta}_{F@t+1}$ takes the seven inputs of ANN_y as well as $\vec{y}_{H@t}$. For the $n > 3$ case, we explain how coordinate transformations allow these ANNs to be applied recursively to predict all robot positions in the following section.

2.5.1.2 Scalable, Modular Prediction Strategy

For estimation with $n = 3$, the ANNs on *Tail* can estimate $\vec{p}_{H@t-1}$ and $\theta_{F@t}$ at any time t using only the locally sensed position of *Follower* (assuming the initial orientation of all robots are known). Throughout the estimation process, the $\vec{x}_{H@t}$ input to ANN_y and ANN_θ will be estimated by ANN_x, and the $\vec{y}_{H@t}$ input to ANN_θ will be by ANN_y. For example, as the ReTLo technique is needed from time $\tau + 1$, using the position information of *Follower* at τ and $\tau + 1$, the *Tail* can first estimate the orientation $\hat{\theta}_{F@\tau+1}$. Then, by utilizing the subsequent observations of $\vec{p}_{F@\tau+1}$ and $\vec{p}_{F@\tau+2}$ and the predicted orientation $\hat{\theta}_{F@\tau+1}$, *Tail* can estimate $\hat{\vec{p}}_{H@\tau+1}$ as well as $\hat{\theta}_{F@\tau+2}$. Similarly, for every time instant $t > \tau$, *Tail* can estimate $\vec{p}_{H@t-1}$ and $\theta_{F@t}$.

As described in Section 2.5.1.1, our approach trains three ANNs for the $n = 3$ case and applies them recursively to estimate the positions of robots in the $n > 3$ case. An alternative approach would be to apply ANNs trained on data for cases where $n \geq 3$. However, training under that approach would be complicated by a relatively large state space, and a new training phase would be required whenever the number of deployed robots changed.

For scalability and to simplify the training process, we take advantage of a coordinate transformation. Assume that $n > 3$ and *Tail* has processed the estimation until $t = \tau + 3$ as described above. *Tail* can transform the estimates for robots 2 and 3 into the reference frame of robot 2 to predict $\vec{p}_{4@\tau+1}$; that is, the *Tail* can act

as if it is robot 2 predicting the position of robot 4. The predicted position also can be transformed back to the natural reference frame of *Tail* for it to gain the global awareness. By such a repeated computation, *Tail* can finally obtain estimates for the full pose information up to robot $i \in \{3, \ldots, n-1\}$ and thus also the position information up to robot $(i+1)$ in finite time. Estimates of a far robot $i \in \{3, \ldots, n\}$ will be delayed, but this delay scales only linearly with the distance from the *Tail*, which is no worse than would be expected with direct communication of position in a distributed, ad hoc wireless network. Thus, the ANNs trained in the 3-robot case are a general purpose tool for inferring pose information for arbitrary length chains.

2.5.2 Experimental Configurations

Experiments were conducted in the simulation platform designed for use with the *Robotarium* Pickem *et al.* (2016), a remotely accessible swarm-robotics testbed consisting of large numbers of *GRITSBot* robots Pickem *et al.* (2015). The *GRITSBot* is a nonholonomic, differential drive robot that is modeled explicitly within the Robotarium simulator, available in both MATLAB and Python. In principle, robotic implementations tested within the simulator can easily be ported to the actual *Robotarium* multi-robot testbed environment and executed remotely. For our simulated case, the rectangle-shaped arena has width 1.2 and length 0.7, and the diameter and sensor radius of each robot are 0.03 and 0.12, respectively.

2.5.2.1 Training Procedure

In the learning phase, only three robots, one *Head*, one *Follower*, and one *Tail*, are involved as explained in Section 2.5.1.1. They are allowed to interact with each other according to the built-in motion rules described in Section 2.3. During the interactions, all pose information is recorded every time step to later train the three

ANNs on *Tail*. For efficient learning, *Head* is designed to choose its actions from a predetermined set of linear and angular velocity values, leading the team to move forward or make different curves.

In addition to the velocity of *Head*, it is important to vary the motion-rule parameter k_t on *Tail* as well. The ANN is trained in a 3-robot scenario where the motion of *Tail* is not constrained by any robots that follow its own motion; however, when applying the ANN recursively in the $n > 3$ case as described in Section 2.5.1.2, the fictitious *Tail* robots in each recursion step will be more constrained than a true *Tail*. By training with different values of k_t, the ANN is able to anticipate this reduced flexibility.

Our training data is generated from the 20 combinations of the five *Head* velocities with four k_t parameter values. Each of the 20 treatments was allowed to run for 800 time steps, and the resulting trajectories were replicated five times so that Gaussian noise $\epsilon \sim \mathcal{N}(0, 0.001^2)$ could be added to each datum to ensure stable performance of the ANNs even with noise. All ANNs were trained for 500 epochs with a learning rate of 0.05 and momentum 0.2.

2.5.2.2 Estimation Phase

To examine the feasibility and accuracy of our estimation method in various settings, we first show position estimation results from experiments where the *Tail* only has access to local information about its immediate neighbor. In particular, the reliability will tested on various types of team structures while moving, e.g.) straight and curving. Next, a different scenario is introduced where the ANN is used to interpolate between slow updates from periodic communication of other robot positions. Also, we test the effect of noise on the estimation performance.

2.5.3 Quantitative Evaluations

As a measure of estimation accuracy, the *Euclidean distance* between the actual position of robot $i \in \{3, ..., n\}$ and the corresponding estimate in *Tail*'s perspective is calculated every time step. Estimates with an error below the robot diameter of 0.03 are considered to be accurate. Moreover, the testing duration $1,600$ is twice the training duration, which ensures that the ANN is tested with novel challenges.

To differentiate between followers during the experiments, we use the naming convention that *Follower 1* is adjacent to *Tail*, *Follower 2* is adjacent to *Follower 1*, *Follower 3* is adjacent to *Follower 2*, and so on.

2.5.3.1 Estimation without Communication

Fig. 2.2a plots the accuracy in the simplest case where the *Head* leads the other five robots to travel straight. The estimation error is larger for farther robots as they inherit some error from estimations of nearer robots. Yet, every error is below 0.03 and thus within our accuracy tolerance through $1,600$ steps.

We also tested our model with a curve formation that is more complex than a straight line. Fig. 2.2b shows the performance for the case where the *Head* leads a team of five robots to make a curve. The overall error is larger than the previous case, but every error remained in the acceptable level. The error peaked near 400 time steps when the curve's angle was at the maximum during the execution. However, the error decreased after that peak as robots recovered from overshooting the turn and returned to their straight trajectory.

Then, we explored a case of six robots, one more than the previous case, while making a turn. As shown in Fig. 2.2c, the overall performance was worse than the two previous cases, as the estimate for *Head* was inaccurate between 350 and 750 steps.

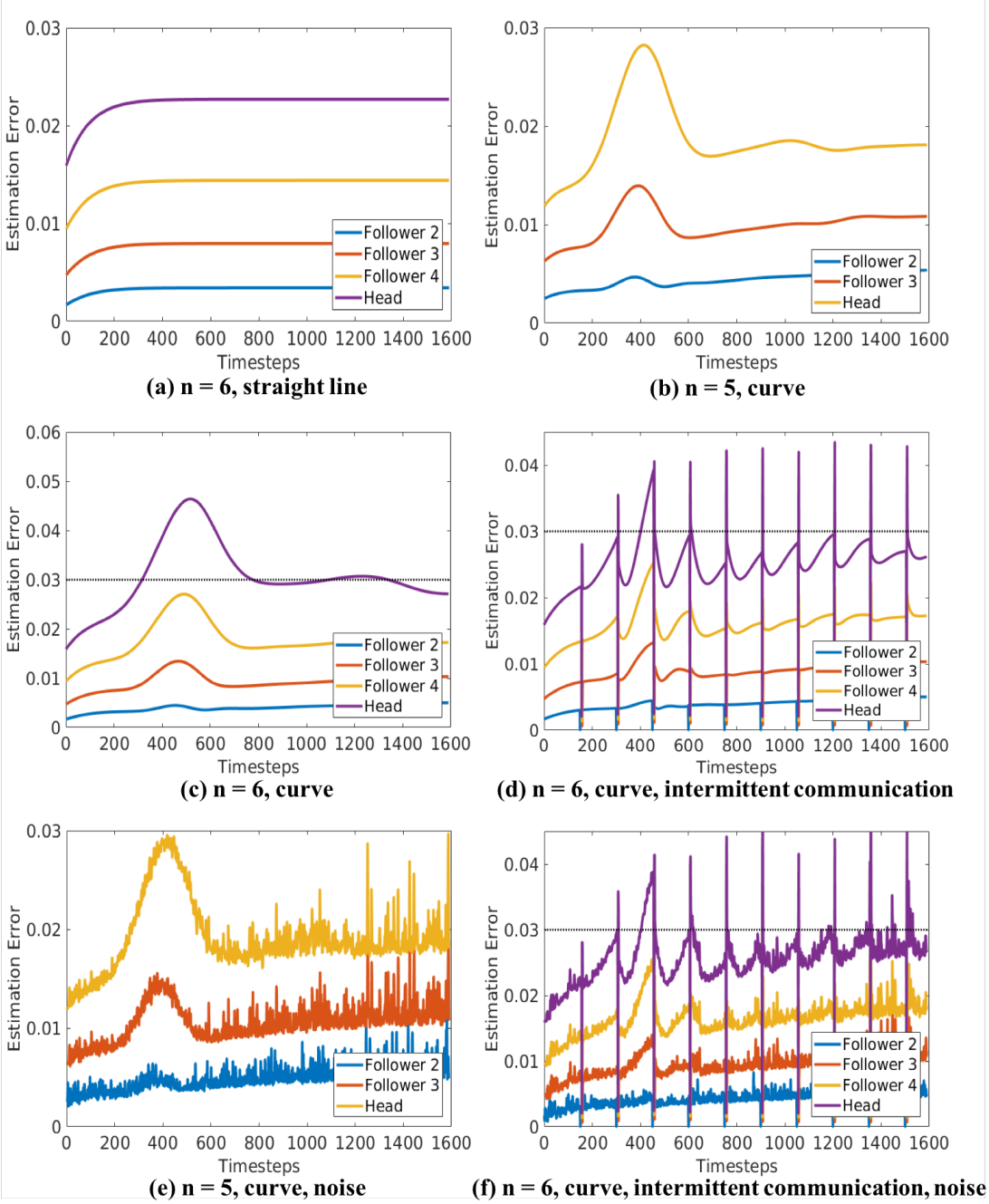


Figure 2.2: Estimation errors. (a), (b), and (c) are cases when *Tail* performs the estimation without any assistance of communication. Cases (d) and (f) allow *Tail* to access to the actual pose information of the team every 150 time steps. (e) is a noise-corrupted configuration of (b), and (f) is of (d).

Though the estimation for the other robots (*Follower 2*, *3*, and *4*) was accurate, the accumulated error caused unacceptable estimation error for *Head*. Nevertheless, the performance pattern looks similar as the previous experiment in that the peak error occurs near 500 steps and declines to below the 0.03 accuracy threshold as the robots return to a more predictable straight-line trajectory after 750 time steps.

2.5.3.2 Intermittent Communication

Although we have focused on scenarios where there is no robot-to-robot communication, we also tested the effect of intermittent communication providing limited information at a relatively slow rate. In reality, we could expect that robots can, in principle, broadcast their own pose data but do not frequently do so due to technological limitations or to save on communication cost. For instance, an aerial robot may only be able to share its pose periodically with a single base station that can relay it later to other robots that periodically fly by.

In our case, every 150 time steps, *Tail* is allowed to access to actual position and orientation information for estimation process. Between messages, *Tail* must use the ANN to infer the pose of others, but actual pose is known intermittently.

Fig. 2.2d illustrates the improvement by the periodic communication, when six robots are turning a curve as previous setting. Just after the communication update, the error rapidly decreases toward zero due to the introduced information from the communication burst. These periodic decreases in error allow for longer periods of time when the total error is kept under the accuracy threshold.

2.5.3.3 Reliability Demonstration

In this set of experiments, we tested the case where measurements were corrupted with noise. Specifically, we added noise $\epsilon \sim \mathcal{N}(0, 0.001^2)$ to every data instance

that is fed to the ANNs in order to examine if the scenarios that already showed successful results would lead to the similar performance in this situation as well. The distribution of noise was chosen to simulate that 99% of measurements would have an error off the true value by ≤ 0.003, 10% of the diameter of simulated robots.

Fig. 2.2e shows a noise-corrupted version of the scenario evaluated earlier in Fig. 2.2b. Despite the additive noise, the ANNs still perform reliably well; every estimate is within the accuracy threshold over time. On average, the error has a similar shape to Fig. 2.2b.

The case in Fig. 2.2f was configured as in the periodic communication case of Fig. 2.2d with additive noise. The noise-corrupted results vary only slightly from the noise-free results, and so our approach can perform well even in a noisy environment.

2.5.4 Use Case: Proactive Assist for Caging Mission

Finally, we return to the motivating multi-robot team application that we introduced in Fig. 2.1b. The multi-robot team is to surround an object after being detected by *Head*. Empirically, we found that caging cannot be accomplished in a team of only *Head* and *Follower* robots because the *Follower* robots have no attraction to the object and thus will not circle it under their simple motion rules. By estimating the behavior of *Head*, the *Tail* can determine when the *Head* has detected an object and roughly where the object is. At that point, the *Tail* can move in a direction that ensures that the *Follower* robots (which are influenced by both their forward and rearward neighbors) are pulled toward the object.

In this scenario, we assume that *Head* and *Tail* can follow along with the surface of an object that they locally encounter in the environment. We also assume that *Follower* uses a potential-field-based controller Arkin (1998) to avoid collisions when in close proximity to the object. Because *Head* will detect the object before *Tail*, we

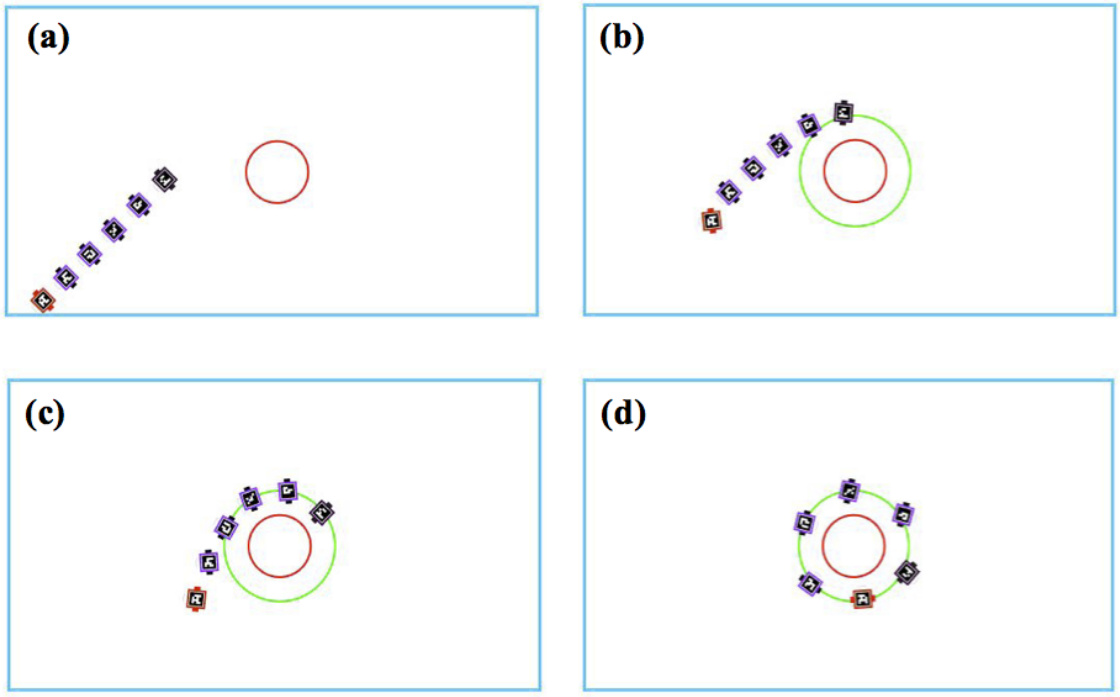


Figure 2.3: Applicable caging scenario for a team with a *Tail* that can infer changes in the behavior of the *Head* from remote.

will study the case where *Tail* detects that *Head* has begun a caging maneuver that can be assisted by *Tail*. In that condition, *Tail* will begin to encircle the object in the reverse direction of *Head*. In the end, when *Head* and *Tail* are facing each other, they will switch to a *Follower* motion rule so that the caging formation converges to equilibrium "net" around the object, in which all the robots maintain the same distance to neighbors over time.

Fig. 2.3 shows our implementation for 6 robots deployed for the caging mission on the simulator. We placed a virtual, circular object for the mission at the center with a circular boundary drawn around it indicating the desired final positions of the robots. The robot team starts searching from bottom left as shown in Fig. 2.3a. In

Fig. 2.3b, *Head* has sensed the object and begins to move around it, and this motion is detected by *Tail* that starts to deviate from simply following its nearest neighbor. 2.3c illustrates how the complementary actions of *Head* and *Tail* lead to an encircling maneuver, and Fig. 2.3d shows the final formation.

2.6 On Physical Testbeds [5]

We have looked at the demonstrations of ReTLo in simulated environments. In this section, the implementation expands to actual, physical robots to investigate the feasibility of the proposed approach on more realistic environments. In fact, the prediction power showed some limitations in previous experiments especially when a longer chain of robots was deployed, and therefore, we might ask questions: "Will the same model be sufficient to perform as well on the real robotic platform, where more noisy factors could be present?", "If not, how could it be improved?" The following sections answer these questions.

2.6.1 Methodology

From the previous method, the fundamental mechanism of "modular" prediction (Section 2.5.1.2 is still used here to take its advantages, but LSTM layers are deployed in the learning model to consider as input signal not only newly incoming observations but also the past trajectory, which includes prior knowledge and previous predictions (c.f. Fig. 2.4). More technical details of the new model are provided in Section 2.6.1.1, followed by experimental settings and results.

[5](Choi *et al.*, 2020)

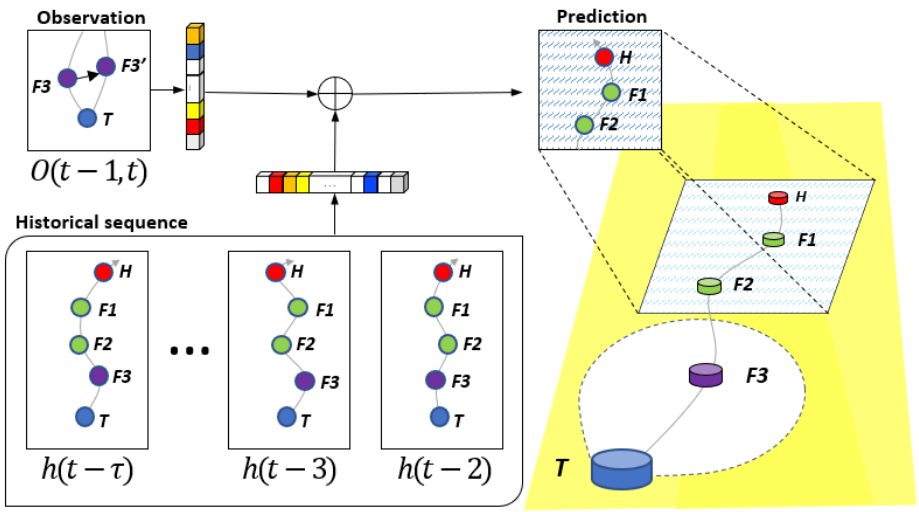


Figure 2.4: Illustration of our proposed pipeline in a snapshot example of 5 robots at time t. Each robot has a limited view and a motion rule dependent on its neighbors except the *Head* robot leading the team at the front. *Tail* uses recent observations on its neighbor, *Follower 3*, which is denoted as $O(t-1,t)$. A sequence of historical poses, h, is also encoded for the model to make a final prediction on the unseen teammates.

2.6.1.1 Learning from Behavioral Sequences

Because the learned regressor is designed to work in a modular team of 3 robots, notations for robot $r \in \{H, F, T\}$ only indicate the identities of *Head*, *Follower*, and *Tail* and not any additional intermediate *Follower* robots. Recall that all positions noted here are assumed to be expressed in the reference frame of the *Tail* robot though we use absolute direction for orientation.

In this work, we use a significantly different deep neural network architecture than our previous ReTLo model on simulations. As shown in Fig. 2.5, our proposed model here uses not only the current observation of the follower as an input but also a historical sequence of poses. We deconstruct the architecture here.

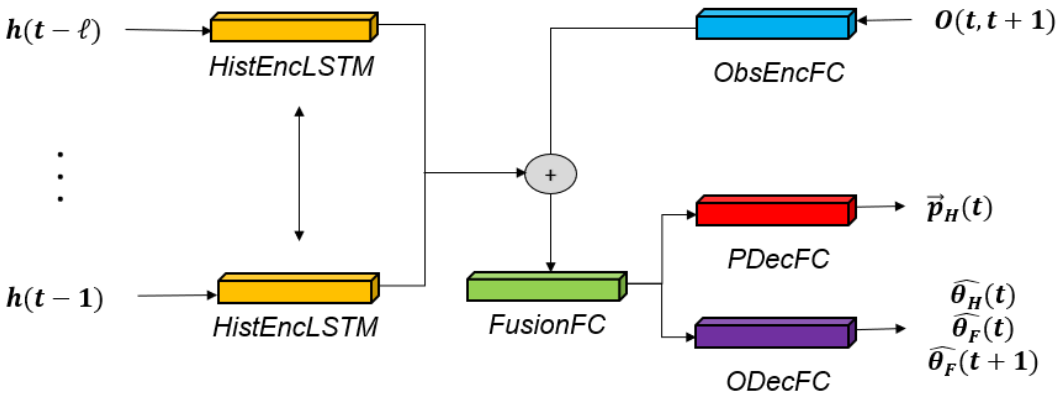


Figure 2.5: Structure of our proposed deep neural network. This is an snapshot example when applied to a focused module of 3 robots, called *Tail, Follower, Head* within it, at time $t+1$. The Encoder–Decoder structure encodes: 1) historical positions and orientations of *Follower* and *Head* until $t-1$, and 2) the observed positions of the *Follower* at t and $t+1$. The decoder part learns to estimate: 1) the position of *Head* at t, and 2) orientations of *Follower* at t and $t+1$ and *Head* at t.

- We use a fully connected (FC) layer f_{obs} of size $k \in \mathbb{N}$ to encode observations into a feature vector $o \in \mathbb{R}^k$. That is,

$$o = f_{obs}(X_{obs}) \tag{2.2}$$

  where $X_{obs} = (\vec{p}_{F@t}, \vec{p}_{F@t+1})$ is the observed positions of *Follower* at times t and $t+1$.

- A historical sequence of poses is encoded by a bi-directional LSTM layer (Wu *et al.*, 2016),

$$h = f_{hist}(X_{hist}) \tag{2.3}$$

  where f_{hist} is a LSTM layer, $X_{hist} = (\vec{p}_{H@t-\ell:t-1}, \theta_{H@t-\ell:t-1}, \vec{p}_{F@t-\ell:t-1}, \theta_{F@t-\ell:t-1})$, and $h \in \mathbb{R}^{2\times m}$ is the encoded history feature where ℓ is the length of historical sequence, and m is the size of LSTM layer.

- The o and h feature vectors are synthesized by a layer ϕ, which is passed as input to two separate final regressors, g_p and g_θ, such that

$$\begin{aligned} Y_p &= g_p(\phi(o,h)), \\ Y_\theta &= g_\theta(\phi(o,h)) \end{aligned} \tag{2.4}$$

  where $Y_p = \hat{\vec{p}}_{H@t}$ and $Y_\theta = (\hat{\theta}_{F@t}, \hat{\theta}_{H@t}, \hat{\theta}_{F@t+1})$.

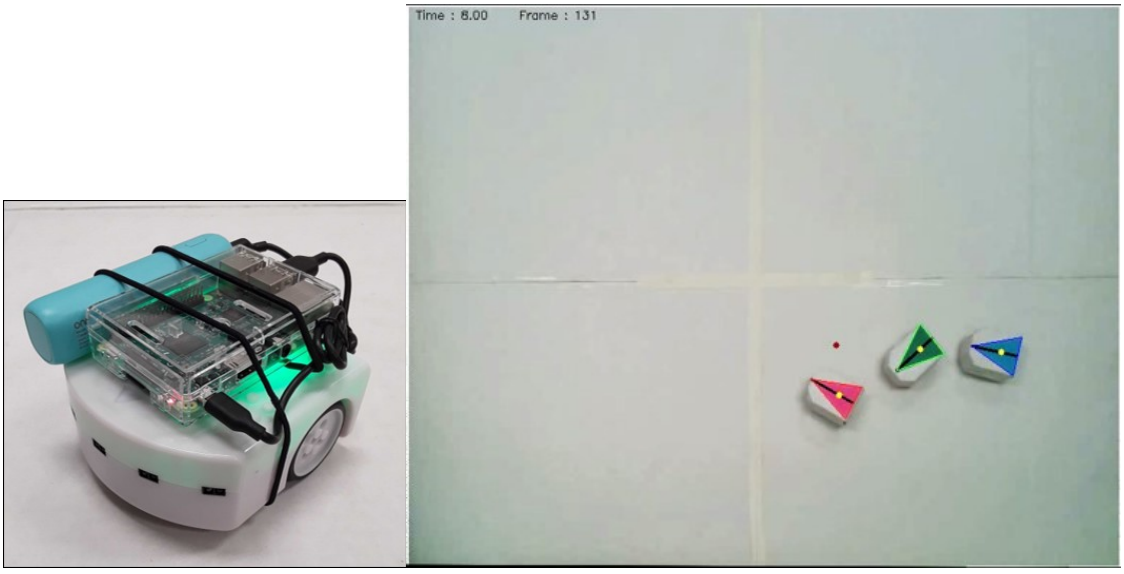
  For fusion of o and h features, layer ϕ in Equation 2.4 could be implemented by any type of layer. During our experiments, we built a FC layer of size $d \in \mathbb{N}$ to find a nonlinear relationship between the input features and achieved a satisfactory performance.

  Furthermore, orientation estimate $\hat{\theta}_{F@t+1}$ gained with $\hat{\theta}_{F@t}$ is estimated again when $\hat{\theta}_{F@t+2}$ is estimated at the next prediction step. Although our model keeps the later estimate only, we discovered that involving it in both steps can regulate the regressor during learning to produce a model that achieves a better validation score.

To find a best combination of model parameters mentioned above, we performed an extensive random search with choices of other learning parameters and finally set $k = 80, m = 160$, and $d = 160$.

2.6.2 Experimental Configurations

Here, I describe detailed configurations that could reproduce the experimental results to be discussed. Because the distinction from previous settings is to perform the macrostate estimation on real robots, the environmental information of employed robotic platform, *ThymioII*, and other connected hardware devices are first explained, and then, strategies for effective data collection are described with the entire training procedure.



(a) (b)

Figure 2.6: (a) *Thymio* robot equipped with a *Raspberry Pi* and a potable battery to stably communicate with the central computer during experiments. (b) Arena of $2.5m \times 1.9m$ observed by an overhead camera on which each robot is covered by a triangle pattern of a unique color to easily track its position and orientation for data collection. The red dot on the arena is the randomly sampled destination of *Head* (Pink robot).

2.6.2.1 Robotic Platform – ThymioII

To demonstrate the effectiveness of our method, we employ a commercially available, two-wheeled physical robotics platform, *ThymioII* (Riedo *et al.*, 2013), which allows to execute a team of small two-wheeled mobile robots. Although each *Thymio* robot has sensing and computation capabilities, we used a central computer connected with a overhead camera to simulate better proximity sensors, more powerful computing power, and a GPS system that would be generalizable to other robotic platforms run in a similar physical scenario. Specifically, the central system is config-

	Duration	Num. of Samples	Num. of Instances
3 robots	100.6 minutes	6,975	465
5 robots	45.0 minutes	8,736	208

Table 2.1: Description of data collected from executions of 3-robot and 5-robot teams.

ured to detect the locations of robots in real time using off-the-shelf computer vision packages and communicate with a *Raspberry Pi* board (Upton and Halfacree, 2014) mounted on each robot (Fig. 2.6a), which implements control laws based on the received positional information about neighboring robots. The *Tail* and each *Follower* robots were configured, as in our previous simulation work, to regulate distance with the robot ahead of it. Regulation of the distance behind each *Follower* robot was achieved through stopping (as opposed to backward motion), which reduced higher frequency oscillations in individual robot motion. Additional details about the physical constraints of the laboratory testbed are can be viewed in a supplementary video at `https://youtu.be/Ok7Car0IcCE`.

2.6.2.2 Data Collection

We collected the pose data from two robot teams, one of 3 robots and one of 5 robots, that ran separately in an arena of $2.5m \times 1.9m$ (Fig. 2.6b). The location detection was performed at 4 frames per second at each of which a new command was received by each robot. Also, all pose data was collected at the rate of 2 frames per second, which was not necessarily synchronized with the command timing. We set the length of history to 5 seconds (10 time steps in data recording) and the time window for prediction to the next 8 seconds (16 time steps).

We also designed a central trajectory planner Ψ to generate highly arbitrary poses

of the robot team without frequent human intervention. Ψ essentially provides random waypoints to the *Head* robot simulating a virtual rectangular grid G of 12×8 cells overlaid on the physical arena, denoted as $c_1, c_2, ..., c_{96}$ where each cell is considered to be networked with other neighboring cells in 8 directions. Ψ particularly operates an array of memory M to maintain indices of the two cells most recently visited by the *Head* (i.e., the latest at $M[0]$ and the earlier at $M[1]$). Immediately after the *Head* has reached its destination, for calculating the next waypoint, Ψ first determines a set of candidate cells $C = \{c_i \mid A(M[0], i) \wedge \neg A(M[1], i) \wedge i \neq M[1]\}$ where $A(a, b)$ is the function returning either $True$ if c_a and c_b are adjacent in G or $False$ otherwise. Then, a random coordinate is eventually drawn as the next destination by uniform distribution across all the regions lying in C, as shown as red points ahead of the moving team in Fig. 2.9 and the supplementary video, linked above. With the wheel speeds in our implementation, the *Head* robot appeared to trigger roughly 2 to 3 times of destination change during the inference time period of 8 seconds.

Table 2.1 provides details about the collected data, where a *Sample* refers to a set of coordinates and orientations in a 3-robot group with which a prediction can be performed, and an *Instance* is a set of all available samples for 13 seconds from the entire team. To reduce temporal autocorrelation to help ensure independence of instances, we clustered recordings so that instances are separated by at least 7 seconds. All the collected data is open to the public to encourage more future works on ReTLo at `https://github.com/ctyeong/ReTLo`.

2.6.2.3 Training Procedure and Evaluation Protocols

Our model is implemented in the *Tensorflow Python* library [6] to realize the entire pipeline, and it was trained to minimize loss functions such as Euclidean distance and

[6] https://www.tensorflow.org

mean absolute errors for position and orientation estimation, respectively. 60% data from the 3-robot team is used to train our model, and another 10% was set aside for validation. At each epoch of training, a validation followed so that the learned weights that achieved the best validation performance were saved. The rest of 30% data and all the data from 5-robot were used to test the model. We compare our proposed model to two different alternative approaches:

- $2X$ *Heuristic*: The prediction on *Head* within a modular subteam is performed by doubling the vector $\vec{p}_F - \vec{p}_T$.
- *FC*: Two fully connected layers run in the predictor without historical information, which is based on the previous model on computer simulators.

2.6.3 Quantitative Evaluations

Here we explore the performance of proposed model in quantitative manners. As in Section 2.5.3, For each prediction, the Euclidean distance from the true position is treated as the corresponding error.

2.6.3.1 Overall Performance

In Fig. 2.7, we compare the averaged error for position predictions of the three models in both the 3-robot and 5-robot team cases. In the case of 3 robots, the average error is calculated only on the prediction for the *Head* robot, while in the 5-robot case, it involves all predictions for *Follower 2*, *Follower 3*, and *Head*. Moreover, for every model except the *2X Heuristic*, the mean performance of 5 separate learning sessions is obtained with the standard deviation. Fig. 2.7 shows that the machine learning methods outperform the *2X Heuristic* in any case. Particularly, the performance gap between *2X Heuristic* and *FC* becomes much larger in 5-robot case implying

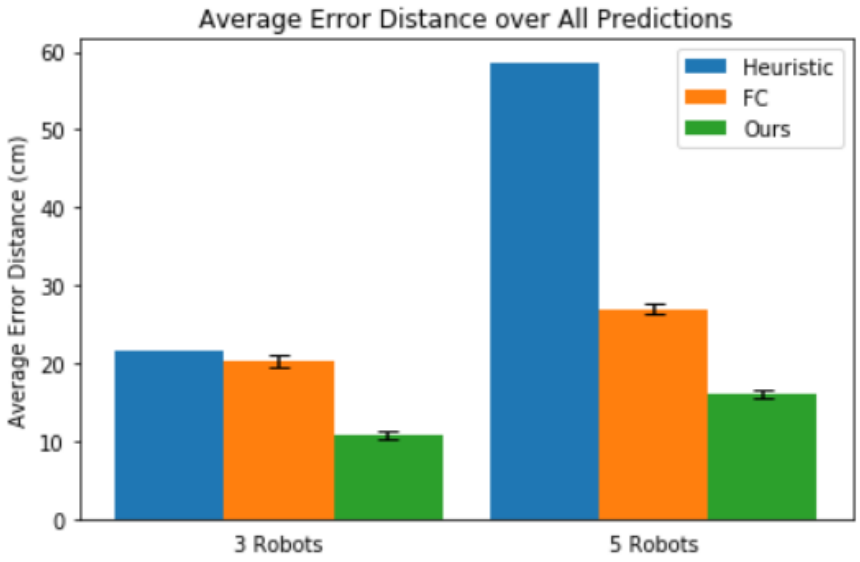


Figure 2.7: Average accumulated error of each model in two different sizes of robot team. For each machine learning method, the mean performance of 5 separate sessions is reported with a error bar of performance variation.

that realistic stochasticity presents a significant challenge to the straightforward *2X Heuristic* case. Moreover, our model clearly shows the performance gain over the *FC* model, since the average error was reduced by 47% and 40% in the 3-robot and 5-robot case, respectively. Also, considering the diameter of a *Thymio* robot is 12 cm, as only three robots are deployed, the average distance between the prediction and the true position is shorter than the length of the robot body. This overall result proves the effectiveness of encoding a sequence of historical behaviors as a feature input over the model fed only with very recent observation.

2.6.3.2 Temporal Analysis

In this section, we analyze the performance of the machine-learning based models from a more fine-grained perspective. For each model, Fig. 2.8 visualizes the step-wise error for different target robots (i.e., *Follower 2* and *Head*) in 5-robot case, which

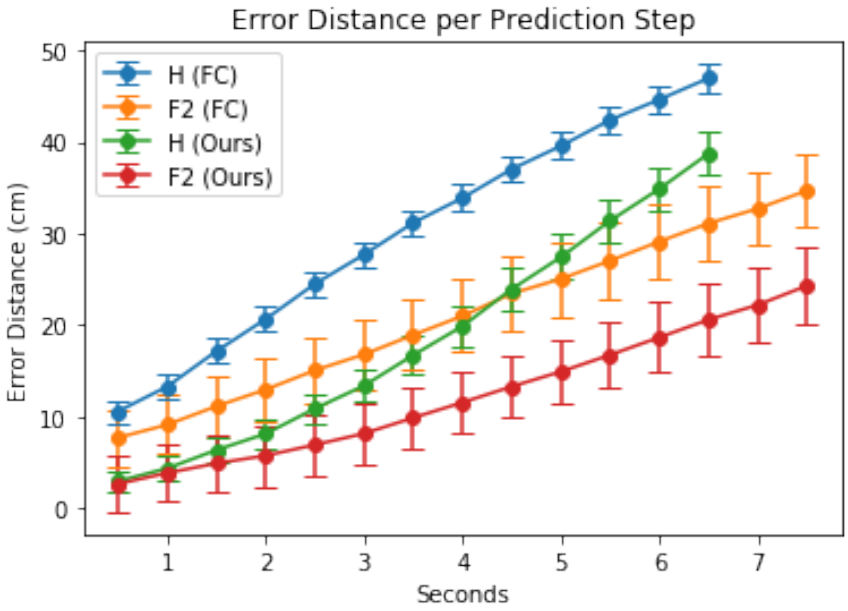


Figure 2.8: Average step-wise error for different target robots in 5-robot team. For each model, all the prediction errors at different time steps are averaged for a specific robot. The error bars represent the standard deviation of 5 separate models in terms of the performance metric. For the sake of visualization, the error for *Follower 3* is omitted, but note that in any model, the error ranges between the two visualized errors at every step.

is calculated by averaging the errors of each step across instances. By doing so, we can better understand the approximate time frame within which estimation error stays below an acceptable error level. During this evaluation, we also had 5 separate training sessions to gain the mean and the variability of the performance. Fig. 2.8 shows that for any target robot, the error increases over time due to the design of the repetitive prediction method where previous errors would negatively impact the prediction power for the following steps. In a similar sense, each model brings about larger error on the *Head* prediction than on the *Follower 2*. However, at every time step, our approach causes less error than the *FC* model for any robot, and the

visualized error bars confirm that the performance improvement of our approach is significant in every case. Especially for first 4 seconds, the prediction on *Head* from our model appears more accurate than the prediction on *Follower 2* from the *FC* implying that until then, our method also has less error for the estimate of *Follower 3* as well. Having said that, the more rapid increase in error of our method for the position of *Head* may highlight the liability of relying more on prior information for future predictions. Nevertheless, these results demonstrate the potential of our approach to reduce the communication bandwidth necessary for a multi-robot team to perform localization.

2.6.4 Qualitative Evaluations

Fig. 2.9 shows images of three different instances of 5-robot team in which both the true positions of all robots and the predictions on them are represented. The triangles indicate the true positions and orientations, where the yellow triangle is the *Tail* and the red triangle is the *Head*. The circles indicate the estimated positions of the two farthest *Follower* robots as well as the *Head*.

Predictions until 4 seconds appear reliable, although there are some errors while the group of robots is turning with a high angle and when the *Head* changes its destination. Predictions tend to be biased more toward the inside of the curve the team is moving on. Unexpectedly, at any instant of time, the maximum position error is often at intermediate *Follower* robots and not at the most distant *Head* robot. Such corrections may be the result of using historical information to constrain the possible locations of farther robots, as in data fusion techniques that use formal dynamical models to physically constrain estimator variance. However, our approach does not utilize explicit formal dynamical models.

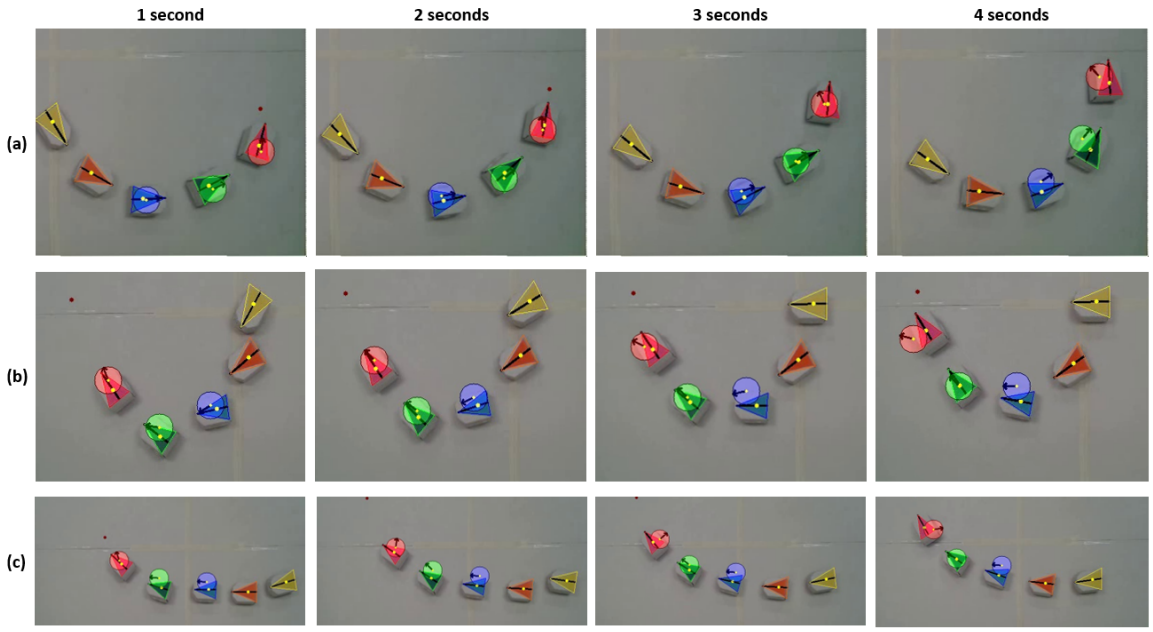


Figure 2.9: Sample video frames with prediction results of our proposed model for three different instances of 5 robot team. Each row presents four frames of an instance for first 4 seconds. The yellow robot is *Tail*, the pink is *Head*, and in-between are *Follower* robots. The colored circles are predicted position outputs, in each of which a black arrow is drawn to indicate the predicted orientation as well. The colored triangles with a black line through the center are the results of localization detection used in data collection stage. A small red dot on the arena in each frame is the random destination the *Head* is moving toward, which is re-sampled at intervals.

2.7 Concluding Remarks

To summarize, we have explored ReTLo problems as a useful example of *macrostate estimation* in complex multi-robot systems. We have designed a scalable algorithm to enable an individual robot in multi-robot system to predict the evolved formation of all teammates, "macrostate", from the motions of an only observable neighbor, "micro observations", while remote communication is limited. Particularly, a *mod-*

ular prediction scheme has been introduced to build practical estimators under the challenges such as **C1∼C3** in Section 1.3 by tuning the model within a modular team of 3 robots and later applying to all other modules when deployed in a larger team – i.e., additional training of model is not needed even when the team size has changed.

We first used a realistic robotic simulator with noise and nonholonomic kinematics to show that a robot at one end of a chain could successfully estimate the position of all other robots with very little explicit message passing. Although performance of the estimator was strongest when the formation took on simple shapes, like a line, the method was still able to estimate robot positions for more complex maneuvers. We showed that the method could allow robots to coordinate to encircle an encountered object and that the far-robot position inferences were sufficient for the robots to coordinate their actions.

In addition, by incorporating historical data into the design of a scalable, localization regression module, we have improved upon our previous simulations in estimating the positions of robots in a multi-robot team using only observations of a single, nearest neighbor robot. The new approach greatly reduces estimator error, giving it the potential to greatly reduce communication bandwidth in other localization schemes that otherwise require continuous communication among robots.

To test our approach, we utilized a commercially available robotic platform, *ThymioII*, to collect datasets from 3-robot and 5-robot teams. Through empirical experiments, we showed that the proposed machine learning model offers more accurate estimation than other approaches. We analyzed our estimator performance over time and demonstrated its improved performance for short horizons but potential added liabilities for longer time horizons. Lastly, we overlaid visualizations of the prediction outcomes on images of the actual robotic system to illustrate specific cases where low estimation error was provided for far-away robots despite high error on

more nearby robots.

Chapter 3

ACTIVE INTERACTIONS FOR NONE-BIASED LEARNING [1]

With earlier ReTLo problems, we have explored the feasibility of predicting formational macrostates from microlevel behavior observations in the system that accompanies some of the challenges mentioned in Section 1.3. In particular, as summarized in Section 2.7, the proposed estimation framework could address **C1∼C3** largely by *modularizing* the learning and application of estimators. However, we may also need to consider **C4** since the training data collected from a modular team of 3 robots might be *biased* toward more limited formational features than the ones that could actually evolve from the whole team. In fact, as stated in Section 2.5.2.1, we have adopted a heuristic approach during data collection to combat such an issue in which constant values such as k_t are controlled for the *Tail* robot to experience varying distances to its neighbor, and it appears helpful in experiments while far teammates tend to lead various amounts of space to their neighbors. Yet, that heuristic has a limitation in that it is "system-dependent" because the motion rule and the controllable values should be known in advance.

In this chapter, therefore, we will discuss an alternative and "system-independent" approach to guide the process of data collection to acquire *less biased* datasets and ultimately gain a better generalizable estimator. To be specific, inspired by *active learning* paradigms (Settles, 2009), we build an additional module on the *Tail* to turn an "active" mode on to purposely take trajectories that would lead *novel* formations during data collection. This is actually a completely different approach than the previous, where the robot "passively" followed a pre-programmed motion rule for

[1] This chapter is based upon (Choi and Pavlic, 2020).

both training and test. This chapter, thus, will provide an opportunity for us to find whether and how the intervention in robot dynamics at the stage of training could bring a significant impact on the learning performance of macrostate estimator.

3.1 Introduction: *Learning is not a one-way process*

Collecting a sufficiently large dataset is essential to fully utilize machine learning algorithms by training on *non-biased* datasets; the trained model should be exposed to a diverse number of scenarios to ensure good performance with arbitrary scenarios that may occur when deployed in the real world. In robotic systems, such a large amount of data can be gathered by repeatedly executing particular motions of robots or demonstrating human actions for specific tasks so that the trained model can successfully understand the involved dynamics from obtained observations. The data generation, however, is not cost-free; most robots are powered by an electric battery, and human assistance may be needed when the robot unexpectedly fails during operation (e.g., humanoids falling down or mobile robots getting stuck on obstacles). Thus, developing an efficient learning method is crucial to lead the trained robot to reach a satisfactory performance with exposures to less but representative data.

In this work, we develop a method for a robot in a multi-robot team to actively choose behaviors that are likely to generate better quality, non-biased training data, thus leading to improved performance with less data gathered. We focus on the Remote Teammate Localization (ReTLo) problem introduced in the previous chapter, where a robot in the rear of a convoy uses passively obtained information from the robot ahead of it to localize all other robots in the convoy without requiring communication by explicit signaling among robots. We showed that a robot trained in a 3-robot team could accurately predict positions of robots in larger teams through repeated application of the trained predictor. Here, we propose the Selective Ran-

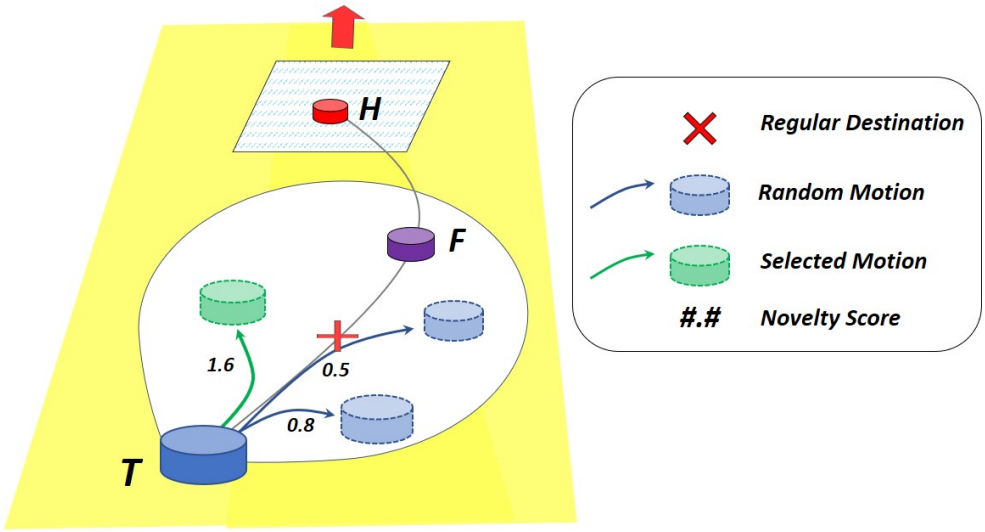


Figure 3.1: Illustration of the SRS framework during data gathering in the 3-robot convoy, as part of the learning process in the Remote Teammate Localization (ReTLo) problem. While the *Head* and the *Follower* are moving as defined in their motion rules, the *Tail* instead chooses to follow a highly novel path among random motion selections.

dom Sampling (SRS) framework for a robot learner in its 3-robot training team to choose among noisy actions that will reveal the most informative team-level features in gathered data (Fig. 3.1).

Inspired by selective sampling in active learning (Cohn *et al.*, 1994; Dekel *et al.*, 2012), SRS enables the estimating robot to select among random behaviors to obtain better training samples and consequently gain accurate localization ability with fewer observations. In particular, our method proposes two additional learning modules to determine which random motions to discard or follow in the view of an individual robot. The first module predicts the future trajectory of entire team to infer the effects of each of k random motions of the learner. The second module is an autoencoder to evaluate the novelty of samples with these candidate motions compared to previously

seen data. The motion that provides the most optimal novelty will be selected.

3.1.1 Chapter Highlights

A brief overview of the subjects in this chapter is as follows:

- **Section 3.3** – We will learn how SRS can be implemented in the form of neural network by which a future team trajectory is *hallucinated*, the novelty levels of random actions are predicted, and only the optimal action is finally selected.
- **Section 3.4** – Experimental configurations on *Webots* are shared with the details of data collection procedure, in which a number of sufficiently complex trajectories are generated for reasonable evaluations .
- **Section 3.5** – *Active* approaches demonstrate further improvement over its counterpart from the previous chapter. Clear effects on data distribution are also visualized to better analyze the proposed method in terms of state space search.

3.2 Related Work

Novelty/Anomaly Detection using Autoencoder

The SRS framework operates a neural network as an autoencoder to evaluate robotic pose sequence inputs in terms of originality. The autoencoder's reconstruction error should be higher for inputs that are dissimilar to ones from the training dataset. Richter and Roy (2017) built a deep autoencoder for visual images streamed from mobile robots. If the imaged environment looks novel, the robot is set to exert a safe behavior ignoring all the commands from the trained policy function because it could behave unexpectedly under the unusual data. Similarly, Kerner *et al.* (2019) dealt with multispectral image data sent from the Curiosity rover on Mars. Some of the im-

ages contained non-typical properties in appearance, and so a deep autoencoder was trained to provide useful features from the reconstruction error so that the final classifier could accurately differentiate the images with novel patterns. Similar approaches to using deep autoencoders for novelty detection exist in detecting mechanical (Oh and Yun, 2018), geochemical (Xiong and Zuo, 2016), and traffic (Zhu and Laptev, 2017) anomalies. All of these existing approaches have focused on automated state analysis or perception after all training data was already available. In contrast, we apply autoencoders for novelty detection in the process of a robotic trainee to make choices that collect data for better future learning.

Active Learning for Robots

Active learning has been a crucial research field in robotics because gathering a large volume of labeled data is particularly challenging with robot operation. For example, there has been much research on implementing intelligent robot systems that can ask a human operator to present behavioral demonstrations to successfully complete the given task especially when the robot faces environments about which it is not sufficiently certain (Chao *et al.*, 2010; Cakmak and Thomaz, 2012; Maeda *et al.*, 2017; Chen *et al.*, 2018). In an approach similar to ours, Palo and Johns (2019) demonstrated that a robot active learner can use the reconstruction error of trained autoencoder as the decision variable on whether to ask for additional demonstrations or not. In our work, however, the robot is situated in a multi-robot team, and there is not a specific task where a human operation can suggest the optimal trajectory of state–action pairs. Furthermore, our framework incorporates the active exploration with random search by selectively allowing it to execute random actions, and it could be considered as a model case that is applicable beyond robotic scenarios.

Active Localization

Active localization methods are to ask mobile robots to behave in a certain way for them to gain better awareness of the locations of themselves or a target in spatial space. In this study, humans are not involved for assistance, and active maneuvers are linked to the capability of state perception, which may sound more relevant to our work than the policy learning. With a relatively long history of research (Burgard *et al.*, 1997; Fox *et al.*, 1998), there have been information-theoretic frameworks in which the robot is essentially to move in directions that decrease the uncertainty on localization predictions. More recently, reinforcement-learning algorithms have been investigated to train the robot agent by designing an effective reward function that promotes the best distributions of candidate locations (Chaplot *et al.*, 2018; Gottipati *et al.*, 2019). These approaches do not involve efficient sampling for further learning, whereas our robot becomes able to learn better in subsequent training. Additionally, our work aims more at better understanding kinematic dynamics of the entire team through experiences as a robot member by moving toward or away from its neighbor somewhat arbitrarily.

3.3 Methodology

In the last chapter, we demonstrated that encoding historical sequences plays a critical role to accurately solve the ReTLo problem in real robotic platforms. The same LSTM-based predictive model is used here for localization with a few changes: 1) all the orientations $\theta_{r@t}$ converted to $(\cos\theta_{r@t}, \sin\theta_{r@t})$ to better quantify border values near $-\pi$ and π (Hara *et al.*, 2017), and 2) the Bi-linear LSTM layer replaced by a regular, unidirectional LSTM layer (Hochreiter and Schmidhuber, 1997) as we have found that it speeds up training and does not degrade performance.

Our original approach of local training followed by global execution is still em-

ployed for scalable inference; neural networks are trained only within a 3-robot convoy but can be deployed directly in larger teams by using the predictions for nearer robots as observational inputs for farther robots. Because the current work focuses on a data acquisition method for better training, most explanations only involve the small team of three robots (*Head*, *Follower*, *Tail*). Details about scaling these prediction results to more robots can be found in Section 2.5.1.2.

Formally, the predictive model takes as input the combination of the historical trajectory h_t of length ℓ and the latest observations o_t at time t where

$$h_t \triangleq (\vec{p}_{F@t-\ell-1:t-2}, \cos\theta_{F@t-\ell-1:t-2}, \sin\theta_{F@t-\ell-1:t-2}, \\ \vec{p}_{H@t-\ell-1:t-2}, \cos\theta_{H@t-\ell-1:t-2}, \sin\theta_{H@t-\ell-1:t-2})$$

and

$$o_t \triangleq \vec{p}_{F@t-1:t}.$$

Then, the estimation output y_t is

$$y_t \triangleq (\hat{\vec{p}}_{H@t-1}, \cos\hat{\theta}_{H@t-1}, \sin\hat{\theta}_{H@t-1}, \cos\hat{\theta}_{F@t-1:t}, \sin\hat{\theta}_{F@t-1:t}).$$

Previously, a small team of 3 robots was used to collect training data while the *Head* was moving toward random waypoints, and the *Follower* and the *Tail* were just following the distance-based motion rules. Our approach here is to diversify the training samples through two ordered phases:

- The Random-Only (RO) phase allows *Tail* to draw additive noises $w_t \triangleq (\sigma_{x@t}, \sigma_{y@t}) \sim \mathcal{N}(0, \Sigma)$ to apply to its original destination $\vec{p}'_{T@t}$ that its motion planner actually returns while collecting first m samples. Then, an autoencoder (ϕ, ψ) and a predictive model ζ are trained with the data of m samples for novelty measurement and action "hallucination" (Caley and Hollinger, 2020) (described later).

- Once the training is finished, the Selective-Random (SR) phase begins during which *Tail* makes use of the trained models to selectively perform the most useful motions only while finally collecting $n - m$ more samples.

3.3.1 Selective Random Sampling

Once m samples are collected during the RO phase, the data are used to train ϕ, ψ, and ζ for different goals in a self-supervised manner. Specifically, models ϕ and ψ are encapsulated in the form of a LSTM-based autoencoder in which ϕ is to learn to encode a sequential input as a compact vector, and ψ decodes it to fully recover the original input (Zhu and Laptev, 2017). The input to ϕ is in fact a shortened version of h_t by discarding the poses from the most recent step:

$$h'_t = (\vec{p}_{F@t-\ell-1:t-3}, \cos\theta_{F@t-\ell-1:t-3}, \sin\theta_{F@t-\ell-1:t-3},$$
$$\vec{p}_{H@t-\ell-1:t-3}, \cos\theta_{H@t-\ell-1:t-3}, \sin\theta_{H@t-\ell-1:t-3}).$$

As a result, the entire autoencoder is trained to minimize the reconstruction error $e = ||h'_t - \psi(\phi(h'_t))||^2$ for the input sequence. We later use e as the indicator of novelty as it would be higher when input h'_t is less likely to have come from the data distribution with which the autoencoder was trained (Richter and Roy, 2017; Zhu and Laptev, 2017; Oh and Yun, 2018; Xiong and Zuo, 2016; Palo and Johns, 2019).

Model ζ also uses $\phi(h'_t)$ as input but learns instead to estimate the discarded poses $(\vec{p}_{F@t-2}, \cos\theta_{F@t-2}, \sin\theta_{F@t-2}, \vec{p}_{H@t-2}, \cos\theta_{H@t-2}, \sin\theta_{H@t-2})$ meaning that it predicts the next poses of the entire team from the history of length $\ell-1$. As the ψ and ζ all use the same encoded vector output from ϕ, we train them all simultaneously (Fig. 3.2) so that the multi-task learning can regulate the entire network and lead to better generalized performance.

With the modules trained, the *Tail* then starts the SR phase to gather $n-m$ more

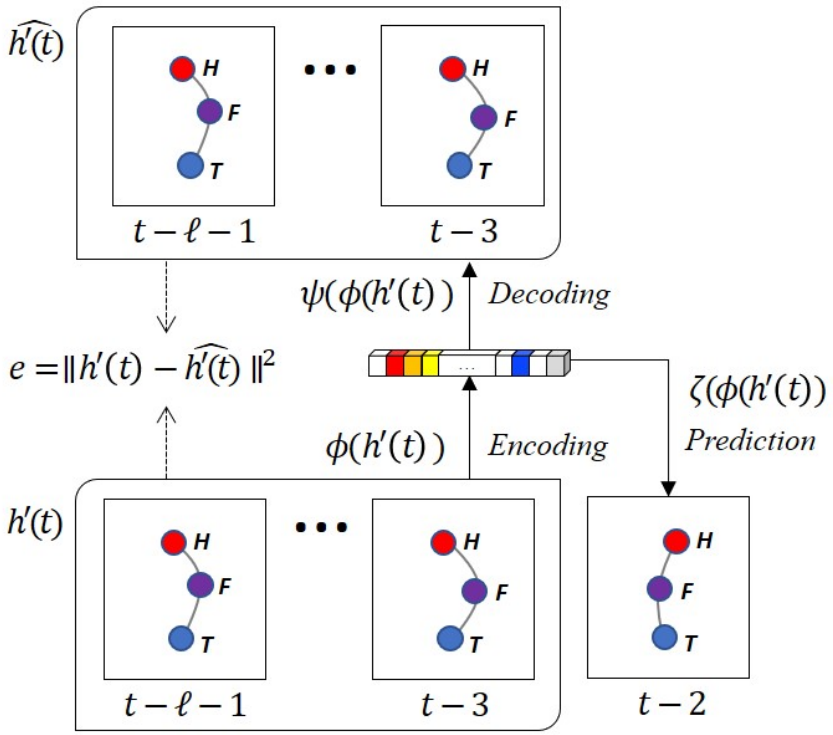


Figure 3.2: Proposed neural network structure to learn the LSTM autoencoder (ϕ, ψ) and the next-step predictor (ζ) simultaneously in a self-supervised manner.

samples. First, the robot draws k random noisy destinations at time t by repeating the process conducted in the RO phase $k-1$ more times. Inspired by the so-called *hallucinations* described by Caley and Hollinger (2020), the (ϕ, ζ) is used to estimate the poses of the robot team at time $t+1$. Then, the robot adjusts the predicted relative positions assuming the feasible trajectory to each of the k random destinations under the non-holonomic constraints. Lastly, the sequence of length $\ell-1$ including the adjusted version of estimation is provided as input to the autoencoder (ϕ, ψ) to compute the reconstruction error e, the novelty index, to select the best motion. Different selection policies are introduced and tested in Section 3.5.2.1 based on the e values of the random actions.

	Team Size	Num. of Samples	Num. of Instances
Train	3 robots	4500	300
Val	3 robots	450	50
Test	5 robots	6300	150

Table 3.1: Description of data collected.

3.4 Simulation Configurations

To evaluate our proposed method in ReTLo scenarios, we simulate *Thymio* (Riedo *et al.*, 2013) robotic platforms within the *Webots* (Michel, 2004) open-source robotic simulator. Robots are programmed to follow at a distance of 30 cm on a simulated arena of 3 m $\times$ 3 m throughout all the experiments. For training and validation data, only 3-robot teams were utilized, while the test datasets involve a 5-robot team as described in Table 3.1. In all datasets, the *Head* is configured to move toward random nearby waypoints that are each centered on a grid cell in a virtual 10×10 grid overlaid on the arena (c.f. Fig. 3.3). A new destination is drawn whenever the robot enters a new cell, as it can move fast enough that 3–4 destinations are given every 10 seconds.

Following our previous conventions, a *sample* is the unit of a set of data with which one prediction can be executed for a focal 3-robot team as the history length ℓ is set to 10 steps. In addition, an *instance* represents a series of samples from the same trajectory for 26 steps when the sampling rate is 2 samples per second, which is the same speed as the frequency of the action command to each robot. Also, each trajectory instance was logged at least 7 seconds after the previous one so that there is little spatial correlation between the instances.

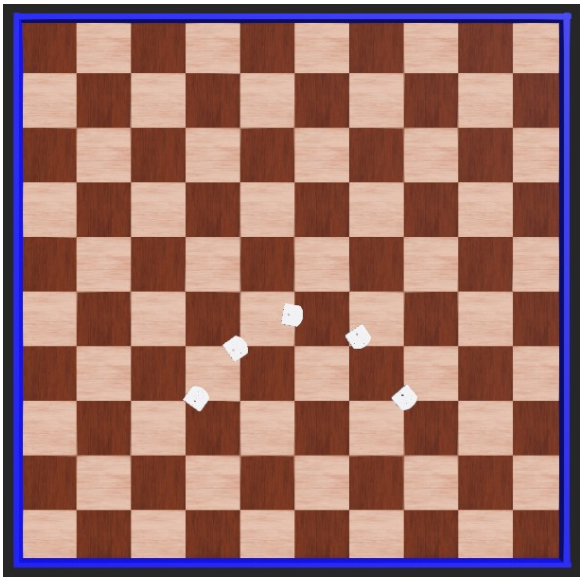


Figure 3.3: Snapshot of the *Webots* simulator running 5 *Thymio* robots on the 3 m × 3 m arena.

Regardless of sampling technique, all evaluated models are set to obtain the same amount of training data from the 3-robot team for comparative evaluation, 300 instances for training and 50 instances for validation. Moreover, in all cases with random motions, we alternate between random and passive modes in *Tail* every 6 steps to avoid completely losing the connection to its team members. Lastly, the accuracy of the models is measured with the Euclidean distance between the true positions and the predictions.

3.5 Evaluations

Here we validate the proposed approach to see whether the SRS strategy can improve the previous model by employing a more active *Tail* robot during data collection. Because randomness is a core property in the SRS, the effect of purely random behaviors are first examined to find any benefit over the previous passive model. Then, we will investigate the properties of SRS itself from the quantified

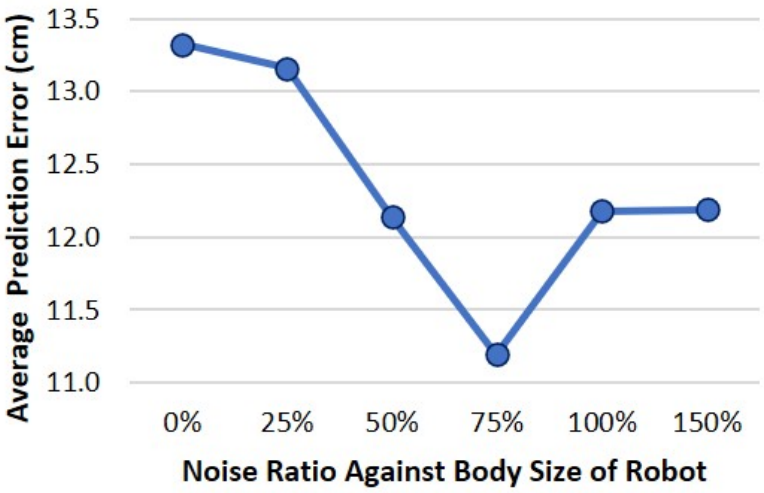


Figure 3.4: Test performance as different levels of random noise perturb the regular destinations *Tail*. *Thymio* size is 11 cm.

performance to the effect to the distribution shape of collected datasets.

3.5.1 Purely Random Motions

We first explore the different noise levels involved in random motions in order to estimate the performance that the pure random policy can achieve. For simplicity, we assume $\Sigma = \mathbb{I}\sigma$ where $\mathbb{I}$ is a 2×2 identity matrix and σ is a predetermined noise level. Figure 3.4 shows the average prediction errors from all predictions on test datasets for six different noise levels (normalized to the 11 cm size of the robot). All noise-levels tested resulted in better performance than the no-noise case, implying that noise injection into the behavior of the tail (which coupled into the behaviors of the rest of the team) may generate more realistic training data. Furthermore, the prediction error is minimized at a non-trivial noise level of 75%; increasing noise further will degrade estimation accuracy. Thus, whereas low-noise levels generate useful training data about the multi-robot dynamics, high-noise levels obscure robot

dynamics. This pattern is similar to the injection of white noise in computer vision to improve the performance of an image classifier. A certain degree of noise can enhance better generalizable features for high-level image classes, but too much noise would likely spoil the key characteristics to the level where a human could not recognize patterns correctly. Hereafter, we use 75% as the noise parameter σ for all the following experiments that involve random motions.

3.5.2 Selective Random Motions

Here, we investigate the performance of our proposed framework in various scenarios when $k = 30$ random noisy destinations are considered at each timestep. We assume the $m = 100$ training instances (33% data) have been gathered already from the RO phase, and the SR phase begins now to obtain $n = 300$ (100% data) in total. To avoid bias caused by random samples, 3 separate datasets are gathered for each configuration resulting in 3 individual prediction models. Thus, the average performance over the 3 evaluations is reported as the final performance.

We first discuss the policies to choose the best motion among the 30 random candidates. Then, the overall and step-wise accuracy of SRS is compared with other baseline sampling techniques:

- Passive: Follow motion rule without additive noise.
- Rand-Only: Persist with the random sampling only.

Lastly, the effects of the SRS framework on the distribution of collected data are explored with visualizations.

3.5.2.1 Selection Policy

We evaluate 4 different policies for selecting among 30 motion candidates that vary in novelty index. Policy *SRS-Max* chooses the motion maximizing the reconstruction error e at each timestep. *SRS-Q2* and *SRS-Q3* pick the ones leading the second and the third quantile of error, while *SRS-Med* uses the median motion.

Figure 3.5(a) shows that as more data is collected reaching 300 instances, the policies that value higher reconstruction errors, such as SRS-Max and SRS-Q3, show a decrease in performance, whereas the other policies increase their performance. SRS-Med finally outperforms the SRS-Max with 100% instances, although is poorer performing with 66% instances. This observation is consistent with the high noise levels in Section 3.5.1. In other words, the prediction model performs accurately because the effect of actions with high reconstruction errors could be balanced off by the original 100 instances from the RO data. Yet, as the high-error samples start overwhelming the training set with more instances, it degrades the overall performance because the training dataset could no longer contain representative features of the dynamic system.

In addition, any policy shows higher improvement than the 12.3 cm error of Rand-Only method when 66% instances are used (Fig. 3.5(b)). This actually implies that the added 33% instances collected by SRS led to the improvement via *selective* random motions and not random motion in general.

Because the SRS-Med is the best policy among others, the following results only involve the model for further analysis.

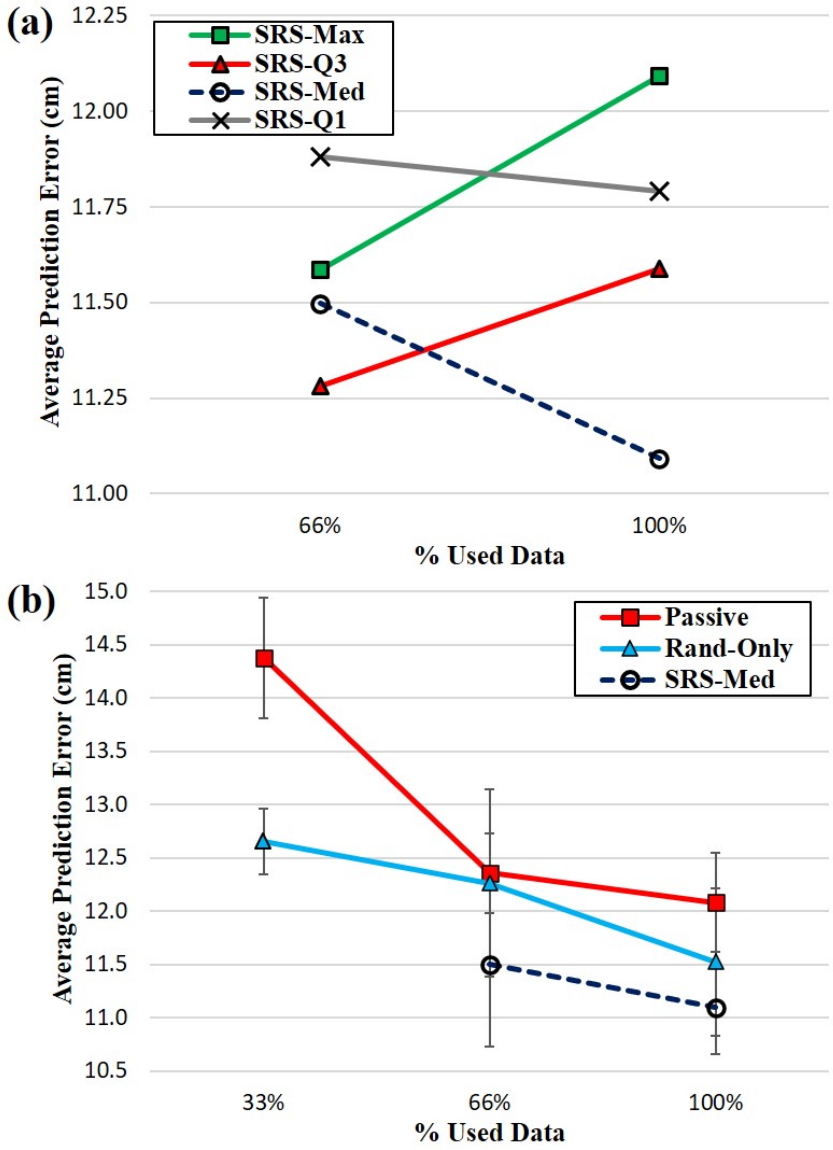


Figure 3.5: Average errors for all predictions in 5-robot teams. In each configuration, the mean performance of 3 individual models trained with 3 different datasets is reported. (a) Four different selection approaches of the SRS. (b) Comparison with baseline models where the error bars represent the standard deviation of individual models.

3.5.2.2 Comparative Evaluations

Figure 3.5(b) clearly highlights the performance margins that our method can gain only with a careful change of training data. As discussed in Section 3.5.1, the Rand-Only method is an improvement over Passive sampling, but the improvement was not as stable as it showed almost a tie with 66% instances. Furthermore, Fig. 3.6 displays that the Rand-Only always offers lower errors for *Head* than the Passive, but it is always less accurate in the case of the *Follower* 2. Thus, although the random motions of Rand-Only in the 3-robot training scenario improve its ability to learn how to accurately predict the position of *Head* in the 5-robot test scenario, the data gathered by Rand-Only is not rich enough to improve predictions of the intermediate robots in the 5-robot scenario.

The SRS-Med approach, however, clearly showed lower error than the Passive as well as the Rand-Only on average. Figure 3.5(b) reveals that even with all available instances, Passive could not reach the performance of the SRS-Med that only used 66% instances. The Rand-Only could achieve that level, but it needed 100 more instances, which suggests that the SRS-based behavior could accelerate learning and save on resources in gathering training data for the same performance level.

From Fig. 3.6, we also observe that the SRS-Med could provide better samples than the Rand-Only in all predictions, though it could not outperform the Passive sampling for *Follower* 2. Still, the performance drop was slower over time to almost reach the same performance level at the end.

3.5.2.3 Distribution Shift of Collected Data

Here, we examine the effect of the SRS algorithm on the distribution of training samples acquired. The SRS-Max with 66% instances is explored to observe the most

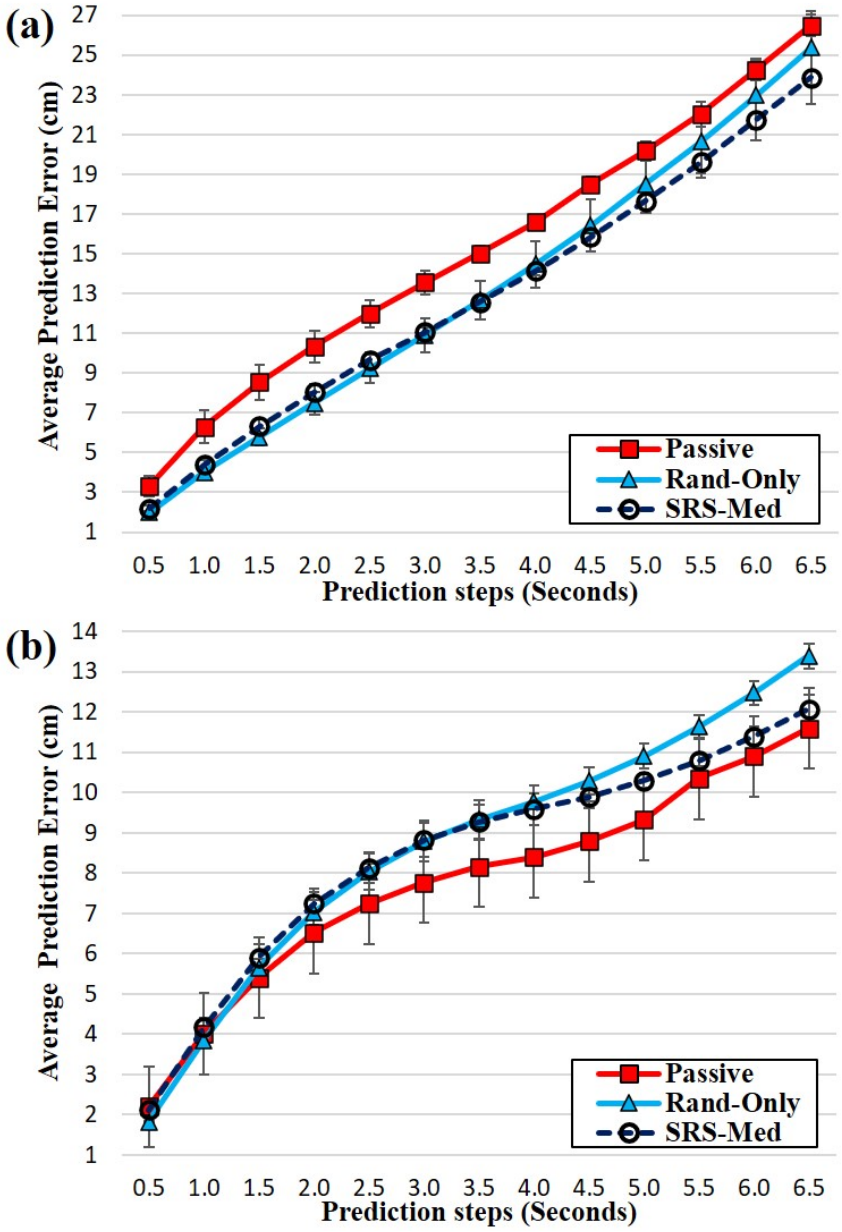


Figure 3.6: Average prediction error of evaluated models in 5-robot teams at each prediction step of tested instances until 13th prediction is performed. (a) Prediction error for *Head*. (b) Prediction error for *Follower* 2.

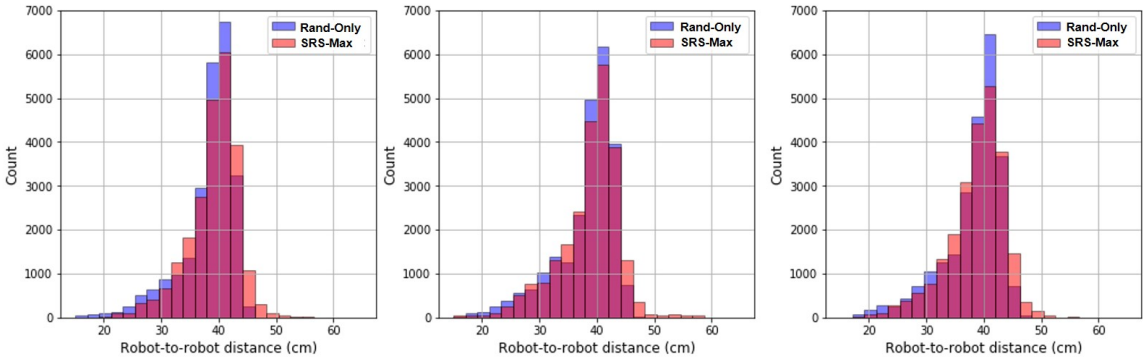


Figure 3.7: Distributions of robot-to-robot distances from three separate 66% datasets collected by SRS-Max. For each gathering, the Rand-Only was applied for the first 33% data (RO phase) followed by the execution of SRS for the next 33% (SR phase).

dramatic change of the sample distributions when the SRS framework is deployed after 100-sample collection in the RO phase.

In Fig. 3.7, histograms of robot-to-robot distance between *Head* and *Follower* and between *Follower* and *Tail* are displayed from three separate datasets. Each histogram shows a bell-shaped distribution centered around 40 cm, which is close to the predetermined distance of 30 cm (consistent with error from a proportional controller). Also, because the Rand-Only and the SRS-Max both have non-random modes at intervals as well, their distributions do not significantly differ. Nevertheless, the histograms suggest that SRS-Max generally picks motions to collect more data with relatively long distances, as the distributions of SRS-Max present longer right tails at the cost of shorter left tails and lower peaks. This supports our design by which the measured usefulness from the novelty detector encourages the robot learner to diverge from its nominal behavior into an unexplored behavioral space that is beneficial for learning more quickly. Consequently, the SRS-Max led to a shift of the distribution of samples toward the side of longer distance where the Rand-Only had gathered sparse samples, to reduce the risk of biased data collection. Finally, the

combination of such distributions can provide better representations for any target robot than using only one of them, as demonstrated in Fig. 3.6.

3.6 Concluding Remarks

As an extension to our previous passive algorithm, we have shown an *active* policy for a robot to better learn ReTLo, so that it can not only maintain the original functionality to tackle **C1∼C3** (c.f. Section 2.7) but also effectively address **C4** from the five big challenges listed in Section 1.3. To be specific, we have discussed effective data sampling methods for an individual robot in a multi-robot system to actively move to be able to access *non-biased* observations for further learning about team-level formational properties.

Our experiments have demonstrated the utility of random behaviors during data collection in learning mobile robots. To a point, injecting randomness into behavior can improve a robot's ability to learn about the coupled system around it. Furthermore, our SRS algorithm takes advantage of an LSTM autoencoder to shape its noisy behaviors so as to be more useful by predicting *novelty* of the hypothetical actions. We have found that there is an optimal level of novelty because the actions of too low would lead similar macrostates to the ones already seen, while too high would extremely peculiar states that would not likely faced in a natural dynamics.

Such a success of the novelty-driven policy, however, may lead the following questions: *"Is involving active agents really necessary in any case of macrostate estimation?"*; *"What if it is too expensive to deploy an active, controllable agent, as in natural social systems?"* In the next chapter, we will use exemplar natural systems as a testbed to pursue the above questions and will devise alternative approaches if needed.

Chapter 4

INTENTIONAL BIASED LEARNING FOR STATE DETECTION IN INSECT COLONY [1]

We have discussed positional macrostate estimation in robotic multi-agent systems – ReTLo problem – in which an individual robot can notice the positions of all of its teammates only from the behavioral variations of the nearest neighbor. In particular, Chapter 3 has introduced controlling the robotic learner to be "active" in motion selection to better gather "non-biased", representative samples about the dynamics of the entire team for training. Although such a strategy has demonstrated a significant improvement on learning performance also addressing **C4** (c.f. Section 1.3), the same approach may not be applied, for instance, in natural systems, where particular individuals cannot easily be controlled. A human or robotic operator could be involved to control a realistic replica to deceive the living mates and promote a certain macrostate (Yang *et al.*, 2019), but it will be very costly.

Through this chapter, we will study macrostate estimation scenarios in which *intentional biased learning* can be effective as an alternative approach to **C4**, so that "active" state space searching is not needed. To be specific, we will employ ant colonies as a representative of natural complex systems (**C1**) from which behavioral observations in "normal" state are only available to learn a model while it must be able to detect whether the hive has turned "abnormal". One-class Classification (OC) paradigms are adopted for a successful binary classifier to build from the *monotonic* observation data, with an assist of a generative model that can synthesize artificial ant motions.

[1] This chapter is based upon (Choi *et al.*, 2021).

In addition, **C5** could be present in natural systems especially if some microlevel interactions are *salient* but cryptic to human observers in predicting macroscopic states. As a potential tool to decrypt the relationship, an existing method *Grad-CAM* (Selvaraju *et al.*, 2017) will be showcased on top of the macrostate estimator to locate specific agents who are engaged in informative behaviors that the model has used in leading its decision.

4.1 Introduction: *Can we learn "abnormal" from "normal"?*

In natural social systems, complex interactions among large numbers of individuals can give rise to phenomena such as "collective minds" (Couzin, 2007) and, as in colonies of ants, even "superorganisms" where it is often more convenient to describe the collective as a single monolithic entity moving from one macrostate to another. Some species of ants have colonies sufficiently small to be observed in their entirety with state-of-the-art video-recording technologies while still sufficiently large to have rich, multi-scale behaviors. Whereas the dynamical processes underlying the non-trivial interactions between ants are cryptic to human observers, there is potential for techniques from machine learning and artificial intelligence to identify social interaction patterns that warrant further study. For example, in species of ants that can elect new reproductive individuals after the previous reproductive is artificially removed (Heinze *et al.*, 1994; Sasaki *et al.*, 2016), machine learning could in principle help to identify abnormal patterns that only occur during this conflict resolution. However, such a behavioral classifier for underappreciated *abnormal* patterns would necessarily be limited to training data from videos of behaviors under known *normal* conditions.

Here, we propose an alternative application of One-class Classification (OC) to solve the *abnormal* state detection problem for video data of social systems under

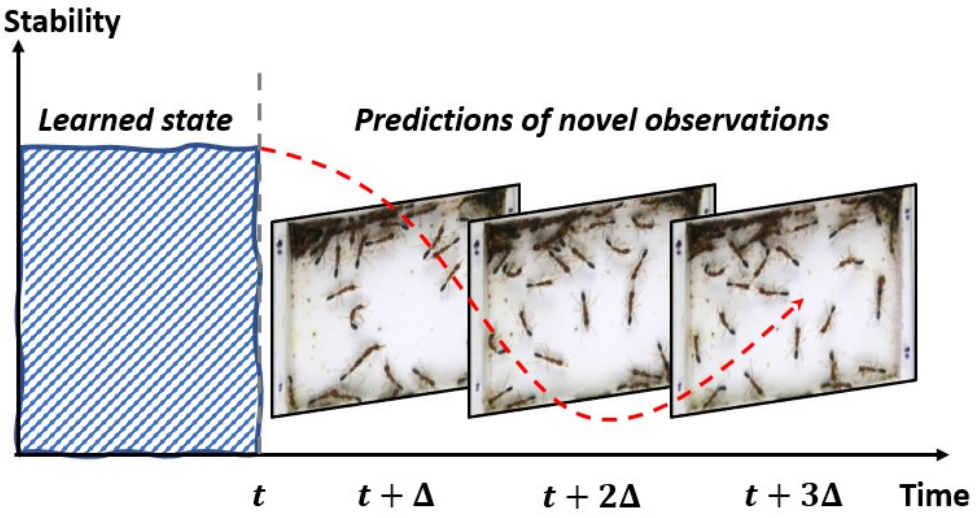


Figure 4.1: Proposed scenario in which observational data from an ant colony are accessible during its stable state, while the trained predictive model is to classify behaviors from unseen, unstable states.

transition. In these systems, behavioral data for training the classifier is only available for the typical state of the system, but the classifier must be able to classify abnormal samples presented to it after training. For OC problems, Support-Vector-Machine–inspired approaches have widely been utilized with the combination of autoencoders, which can learn key features in unsupervised manners while significantly reducing the number of dimensions of original input (Xu *et al.*, 2015; Ribeiro *et al.*, 2020). One of the most successful algorithms of this sort is Deep Support Vector Data Description (DSVDD) (Ruff *et al.*, 2018), which learns a hyperspheric feature space where the samples of an available class lie densely around a central point $\vec{c}$ so that the distance from it is used as the indicator of novelty during test. We argue that the distancing approach of DSVDD may oversimplify useful relationships among features for OC especially in high-dimensional spaces. Thus, we propose a generative module, *Inner Outlier Generator (IO-GEN)*, to replace the heuristic reference $\vec{c}$ with synthesized

"*inner outlier*" observations of an imaginary social-system state so that a separate classifier on DSVDD can use both the hyperspheric structure of data description and high-dimensional feature representation to learn behavioral abnormality.

We apply this IO-GEN approach to the analysis of Jerdon's jumping ant (*Harpegnathos saltator*), which typically resides in a *stable* (normal) state while it has a set of reproducing workers known as gamergates. When the gamergates are removed, the colony moves into a transient *unstable* state (abnormal) during which members go through a process that comes to consensus on a subset of workers that take over the role of the previous gamergates (Heinze *et al.*, 1994). There are known behavioral interactions that only occur during the unstable colony state, but a detailed understanding of how the transient state is resolved remains elusive. We created a dataset for analysis by extracting optical flows from a colony of over 50 *H. saltator* ants to record their behavioral data for 20 days in a lab setting where the colony was artificially triggered to induce "stable–unstable–stable" colony state transitions. We then developed an approach following the simplified diagram in Fig. 4.1. Video data of a particular stable colony is used for normal-class training, and our proposed model then later assesses whether a focal colony is stable or unstable based on a short sequence of new optimal flow inputs.

Lastly, we will move one step further to examine the potential use of our model with Grad-CAM to *decipher* the role of microscopic interactions as the macrostate transitions – i.e., the estimator will be asked to indicate *salient* behaviors that it has associated with particular global states when making predictions. Using the visualized associations, 1) decision makers could better assess the focal social system from observed behaviors, and also, 2) the predictive model could be verified by the experts who can conclude whether the discovered behavioral features look reasonable and compatible with their prior knowledge of the society.

4.1.1 Chapter Highlights

A brief overview of the subjects in this chapter is as follows:

- **Section 4.3** – I will briefly provide background knowledge of unique properties in the species of *H. saltator* to emphasize the value of them as a testbed in this study.

- **Section 4.5** – The details of used dataset are offered, such as the recording process of the 20-day video and the number of samples for training and test. An online link is also shared to foster future studies on it.

- **Section 4.6** – I will explain each component in the proposed pipeline comprised of DSVDD, IO-GEN, and Classifier. Structural information is followed by the specific hyperparameter settings. Program codes are also open to the public online to offer concrete ideas of the implementation. Lastly, the Grad-CAM method is explained to visualize learned behavioral features.

4.2 Related Work

Behavioral Cues for Inferring Collective States

Inference and prediction of current and future collective states is potentially useful in a number of applications. For example, for human crowds, intelligent surveillance cameras can detect abnormal collective states (e.g., conditions consistent with group-level panic or rioting behavior) (Mehran *et al.*, 2009) so that authorities can prioritize surveillance resources and execute proactive mitigation strategies. Alternatively, as introduced through last chapters, individual robots in a multi-robot system can use local information of the pose of nearby robots to infer the large-scale formation of the team it participates in and then alter its own trajectory to more effectively achieve a

group-level response to a stimulus encountered at distal ends of the team.

In behavioral ecology, however, modeling efforts have been focused either on the coarse-grained collective scale or the fine-grained individual scale but rarely the connection between the two. For example, many mathematical and statistical models have been developed for understanding the evolution of group-level states and how they adapt to changes in the environment (Couzin *et al.*, 2002; Reid *et al.*, 2015; Sasaki *et al.*, 2016; Pratt *et al.*, 2002; Buhl *et al.*, 2006). These approaches provide insights into the overall function of collective states but do not provide so much insight into how map observations of an individual to the collective context of that individual. On the other end of the spectrum, more recent deep learning approaches for image segmentation or object detection have been tuned to track individual animals from video frames (Bozek *et al.*, 2018; Nath *et al.*, 2019). These efforts are focused on accelerating data acquisition for existing statistical pipelines that human researchers employ but on making automated inferences across the individual–group scales. Calhoun *et al.* (2019) used an unsupervised learning framework to discover latent states in *Drosophila melanogaster* flies during courtship, but the inference scale was only limited to the group of two engaged flies while our work deals with much larger social groups.

One-class Classification (OC) for Visual Data

Classical OC methods, such as One-class Support Vector Machine (OC-SVM) (Schölkopf *et al.*, 2001) and Support Vector Data Description (SVDD) (Tax and Duin, 2004), either use a hyperplane or a hypersphere tightly bounding the known-class data for separation. Recently, these methods have been augmented with autoencoders that highly reduce input dimensions of graphical data without supervision (Xu *et al.*, 2015; Ribeiro *et al.*, 2020). As an extension, DSVDD is designed to optimize the objective

of SVDD in an end-to-end deep neural network pipeline (Ruff *et al.*, 2018). In particular, the autoencoder's encoder is fine-tuned to generate a feature space in which the in-class samples lie densely close to a pre-defined central vector $\vec{c}$, while the out-of-class samples are sparsely away from them (Fig. 4.2b). Although DSVDD showed

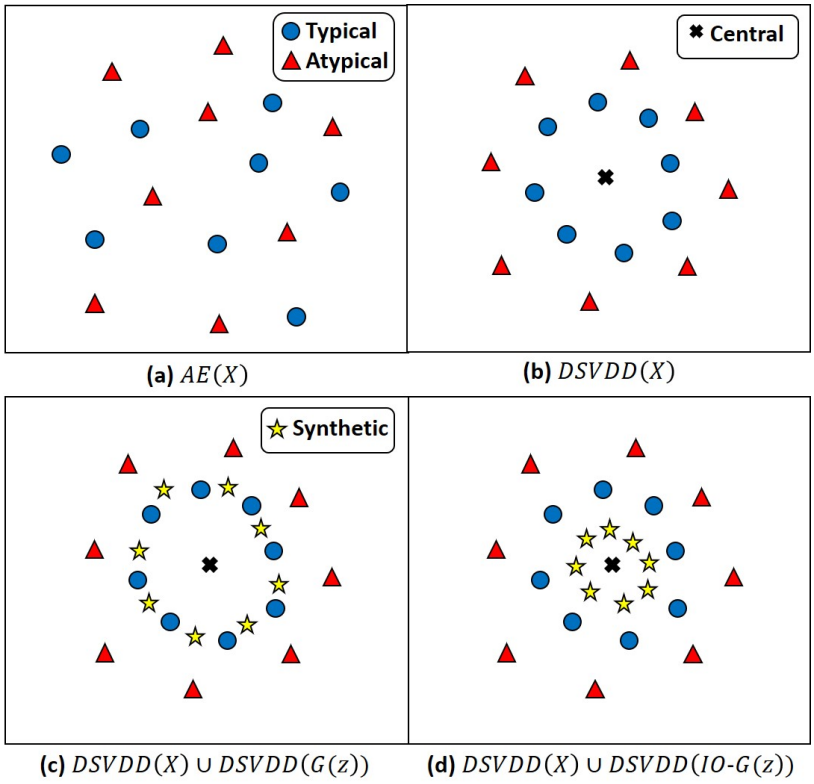


Figure 4.2: Conceptual feature formations in different methods on two-dimensional planes assuming only typical examples have been used for training each model. (a) The encoding from autoencoder cannot ensure a clear separation of atypical samples. (b) DSVDD forms the space in which data of seen class surround a central point $\vec{c}$ more closely than unseen examples. (c) Generators in GANs learn to produce typical properties used for training. (d) IO-GEN synthesizes inner outliers much more densely to later substitute for $\vec{c}$.

competitive performance with several benchmark datasets, we argue that its distanc-

ing scheme – simply using the distance to $\vec{c}$ – is not sufficiently rich to distinguish novel samples, and we combine DSVDD with a generative approach to improve OC for this case.

Generative Adversarial Networks (GANs) (Goodfellow *et al.*, 2014) can also be used in OC by synthesizing fake outcomes consistent with typical samples that are then used in training to improve the generalizability of the OC (Sabokrou *et al.*, 2018; Yadav *et al.*, 2020; Perera *et al.*, 2019). However, these approaches adopt the conventional min–max scheme of GANs to closely emulate the data distribution of the available class (Fig. 4.2c) although the ultimate goal is to identify novel samples from a different distribution. Instead, our IO-GEN generates fake outcomes even closer to the idealized central vector $\vec{c}$ (Fig. 4.2d). These more prototypical samples allow the subsequent classifier to learn sharper discrimination in the DSVDD feature space between normal and abnormal samples.

4.3 Backgrounds

Here, we explore preliminary knowledge about the employed species of ant, *Harpegnathos saltator*, to better understand the social dynamics in their colonies.

4.3.1 Harpegnathos saltator

A key characteristic of ants is their reproductive division of labor. In most ant colonies, a single "queen" is chiefly responsible for laying eggs that develop into "workers" that are responsible for caring for the next generation of eggs. Because workers typically cannot produce new workers themselves, the death of a queen usually means the expiration of the colony shortly after. However, certain species of ants have more flexible workers. In the case of *H. saltator*, workers have retained the ability to mate but do not lay eggs while another reproductive is present. However, when there is no

Figure 4.3: Two examples of *Dueling* interaction, for each of which four sequential snapshots are captured only around the participants. As at the top, largely, two ants are engaged with bouts of active antennation moving back and forth. More ants, however, can participate as well as seen at the bottom.

living reproductive in a colony, mated workers engage in a several-week hierarchy reformation process that terminates as several mated workers activate their ovaries and begin to produce eggs (Liebig *et al.*, 1999; Sasaki *et al.*, 2016). The ascension of these so-called "gamergates" inhibits this process from continuing, thus bringing the colony back to a typical state. When those gamergates die or are removed, this process will begin again, thus allowing colonies to survive essentially indefinitely (Liebig *et al.*, 1999; Sasaki *et al.*, 2016). During the several-week transient hierarchy-reformation state, mated workers can be observed performing special sterotyped aggressive behaviors known as *dueling* and *dominance biting* (Peeters and Hölldobler, 1995; Heinze *et al.*, 1994) (c.f. Fig. 4.3). Although these behaviors are clear signs that the colony is in this transient state, the precise sequence of events that leads to colony-level

resolution is still unclear.

4.4 Abnormal State Detection

Abnormal state detection is a binary classification problem to correctly detect the colonial macrostate as either "stable" or "unstable" given a short observation sequence of ant behaviors. Since we adopt the OC approaches to respect realistic scenarios, we assume that following the notation in Section 1.2, the original state M_t = "stable" as $\tau \leq t \leq \tau'$ where τ and τ' are the initial and the last time stamp of the observed period when the training data is available, respectively. Thus, our goal is to learn the estimator f to accurately predict later state $M_{t'}$ using behavioral observations $O_{t'-\Delta:t'}$ where $\tau' < t'$, and Δ is as a short period of time as several seconds. More details about the composition of observational input and the designs of the estimator are described in the following sections.

4.5 Optical Flow Datasets from Colonies Stabilized

Deep-learning methods for image and video data have shown that optical flows can effectively complement classical RGB data in learning because they can extract transient behavioral characteristics (e.g., shooting), while RGB data largely provides the understanding of scenic context with visible objects (e.g., a bow and arrows) (Simonyan and Zisserman, 2014). Because our framework only expects ants and crickets they feed on in scenes, we only use optical flows in our datasets so that learning will be based solely on behavioral flows; this is a similar approach to the human-crowd behavior classification by Mehran *et al.* (2009).

Figure 4.4 shows the basic environments of our video recording from which optical flows are directly extracted. We used a colony of 54 *H. saltator* ants in a plastic nest covered by a transparent glass. While an overhead camera records the nest, not

Figure 4.4: Over 50 ants are placed to densely live in a plastic arena. Some ants can be unseen as they are present in the foraging chamber led by the tunnel at the bottom side. A few crickets are also visible on this snapshot.

all ants may appear in the scene because some may move to an off-camera foraging chamber led by a tunnel on the bottom side of the nest. Videos were recorded for 20 days, denoted as D-2, D-1, D+1, ..., D+18, where D-0 represents the instant removal of all recognized gamergates between days 2 and 3) to artificially trigger the transient state of the colony. From D+1, we observed frequent *dueling* and *dominance biting* until the aggressiveness almost disappeared on the last several days across the group. A shorter highlight video of the 20-day state progression is available at: `https://youtu.be/eGFQb45QejQ`. By performing downsampling techniques, m sequential optical flows were sampled every 2 minutes, and, for each flow, a pair of horizontal and vertical motional representations in the spatial resolution of 64×64 were extracted from two consecutive frames with an interval of 0.5 seconds. The code provided by Wang *et al.* (2016) was used to acquire $1,333m$ stable-class and $11,984m$ unstable-class optical flows in total. Three unique splits of stable class were prepared to

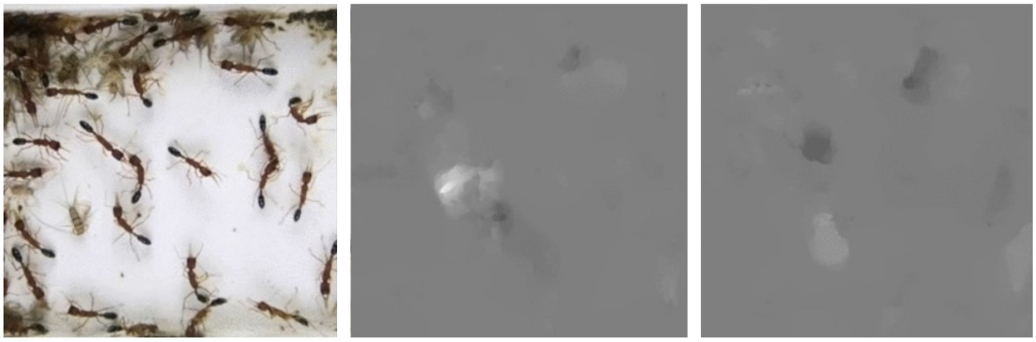


Figure 4.5: Example of optical flows: from the left, the original RGB frame image, and horizontal and vertical representations in which brighter (darker) are more rapid movements in the positive (negative) direction as the top-left corner is the origin.

obtain the average performance of three separate models, as 80% and 20% were used for training and test, respectively in each split, while all unstable samples were included in every test set. All the data and split information are accessible online at `https://github.com/ctyeong/OpticalFlows_HsAnts`.

4.6 Methodology

4.6.1 Deep Support Vector Data Description

DSVDD in our framework follows its original design from Ruff *et al.* (2018). It is built from the encoder part ϕ of a pre-trained autoencoder that us used to learn a feature space $\mathcal{F}$ in which the samples of known class have a lower average distance to a central vector $\vec{c}$ than those of novel class. Specifically, we adopt One-class DSVDD, which minimizes the objective:

$$\min_{W} \frac{1}{n} \sum_{i=1}^{n} ||\phi(x_i; W) - \vec{c}||^2 \ + \frac{\lambda}{2} \sum_{l=1}^{L} ||W^l||_F^2$$

where $||\cdot||_F$ is the Frobenius norm. The first term is closing the distance between $\vec{c}$ and the feature representation of each sample x_i in encoder ϕ parameterized by W,

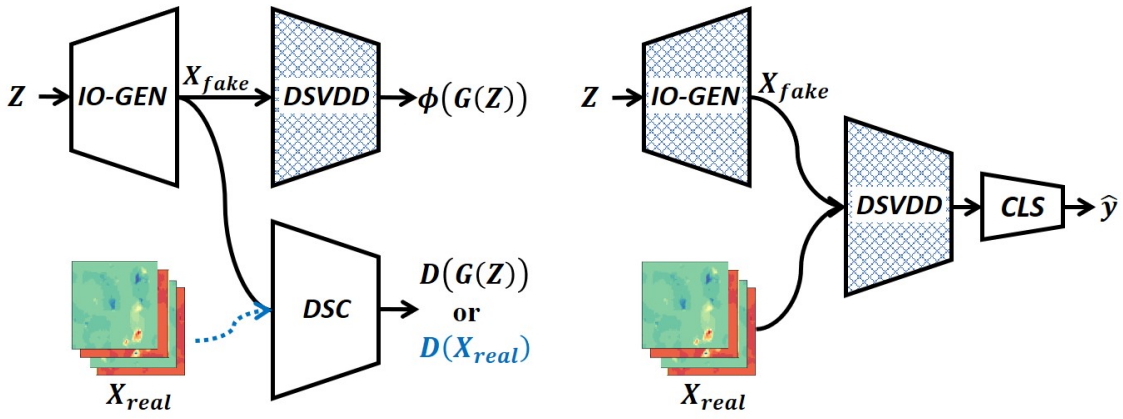


Figure 4.6: Training pipelines for IO-GEN (left) and and Classifier (right). IO-GEN must meet two objectives simultaneously, one with the pre-trained DSVDD and one with the discriminator. Classifier learns binary classification on the data description that the DSVDD offers. Patterned components represent their parameters fixed during the training phase.

and the second term is a weight decay regularizer for L layers with $\lambda > 0$. In the original method of DSVDD, the trained parameters $W = W^*$ are used to generate a distance:

$$s(x) = ||\phi(x; W^*) - \vec{c}||^2$$

that is a proxy for how atypical a sample x is. For some threshold $\tau > 0$, $s(x) > \tau$ classifies x as atypical. Our method, however, substitutes the distancing heuristic $s(x)$ by IO-GEN and Classifier, described below, that we argue better utilize key features in the normal set in order to discriminate abnormal data after training.

4.6.2 Generative Model of "Ideally Normal" Ant Behaviors — Inner Outliers

As shown in Fig. 4.6, IO-GEN G is designed to operate with both the pre-trained DSVDD ϕ and a discriminator network D, as a generative model of optical flows. With the discriminator, an adversarial learning is performed following the standard

objective:

$$\min_G \max_D \Big(\mathbb{E}_{z \sim N_\sigma}[\log(1 - D(G(z)))] + \mathbb{E}_{x \sim p}[\log(D(x))] \Big)$$

where N_σ is the zero-mean normal distribution with standard deviation σ, and p is the probability distribution of real optical-flow data. Due to the first term, the outcomes from IO-GEN are adjusted to appear sufficiently realistic to deceive the discriminator. The DSVDD is used to force the learned synthetic data to be inner outliers close to $\vec{c}$ in $\mathcal{F}$, while the parameters of itself are not updated. In particular, we use the feature-matching technique proposed by Salimans *et al.* (2016), which incorporates the minimization:

$$\min_G ||\mathbb{E}_{z \sim N_\sigma}[\phi(G(z))] - \vec{c}||_2^2$$

The composite loss function for IO-GEN is:

$$L_G = \mathbb{E}_{z \sim N_\sigma}[\log(1 - D(G(z)))] + \lambda \Big(||\mathbb{E}_{z \sim N_\sigma}\ [\phi(G(z))] - \vec{c}||_2^2 \Big)$$

where $\lambda > 0$ is the hyperparameter to determine the relative weights between the two terms to minimize. In other words, IO-GEN is trained to produce behavioral flows of ants that not only look real but also feature the closest proximity to $\vec{c}$ in $\mathcal{F}$ of DSVDD.

4.6.3 Classification Model

Classifier utilizes real data from stable colony states as well as the generated IO-GEN inner outliers to learn to predict the likelihood of unstable behaviors on given m instant frame images as in general binary classifiers. We use a novel strategy, *label switch*, during training by which the real stable samples are labelled as "unstable" (atypical), and the synthetic ones are as labeled as "stable" (typical). This technique leads the Classifier to eventually make low-, mid-, and high-range likelihood predictions for synthetic, stable, and unstable data, respectively, as though the

augmented state of inner outliers was the "most stable" state . That is, Classifier offers likelihood outcomes y somewhat consistent with the class distribution around $\vec{c}$ allowing for a clear separation between real stable and unstable data.

4.6.4 Network Structures & Relevant Parameters

A Deep Convolutional Autoencoder (DCAE) is used as the backbone of DSVDD and IO-GEN once it has been trained with the data of stable ants to minimize the reconstruction error in the Mean Squared Error (MSE) between the input encoded and the decoded output. Per input, m optical flows are all stacked one another to constitute an input $x \in \mathbb{R}^{64\times64\times2m}$ after normalization to range in $[-1, 1]$. In the encoder, three convolutional layers with $32, 64$, and 128 2D kernels are employed in series as each kernel is of 3×3 size. Also, every output is followed by a $ReLU$ activation and 2D maxpooling. The decoder has the reversed architecture of the encoder with two modifications: 2D upsampling instead of maxpooling and an added output layer with $2m$ kernels and a *tanh* activation. Additional 32 convolutional kernels are placed as the bottleneck between the encoder and the decoder to obtain a compact encoding scheme $\vec{v} \in \mathbb{R}^{1\times2048}$ when flattened. DSVDD takes advantage of the pre-trained encoder to reshape the space of $\vec{v}$ by learning data description $\mathcal{F}$ as the reference vector $\vec{c}$ is the mean of available encoded samples according to Ruff *et al.* (2018).

IO-GEN essentially employs a fully connected layer with the $ReLU$ activation that takes a noisy vector $\vec{z} \in \mathbb{R}^{1\times100}$ as input. It is then connected to a replica of pre-trained decoder so that realistic synthesis can be learned faster from the prior knowledge of reconstruction. The discriminator network builds an extra fully connected layer with a *sigmoid* activation on top of encoder, but its weights are all reinitialized because otherwise it appears to easily overwhelm IO-GEN in performance causing unstable

adversarial training. Also, $\lambda = 10$ was empirically found most effective to minimize L_G.

Because Classifier comes after DSVDD, it has an independent architecture in which five convolutional layers learn $8, 16, 24, 48,$ and 48 one-dimensional kernels, respectively. Each layer has a *LeakyReLU* activation ($\alpha = 0.3$) and 1D average pooling, and lastly, a fully connected layer is deployed to provide a predicted likelihood of unstable state via a *sigmoid* function. All codes are also available online at `https://github.com/ctyeong/IO-GEN`.

4.6.5 Salient Behavior Localization

In the last experiment, we will apply Grad-CAM, introduced by Selvaraju *et al.*, on top of the final classifier to visualize the behavioral features learned. Here, based on the original paper (Selvaraju *et al.*, 2017), a brief explanation of the technique is provided with the integration with our framework. Basically, we will utilize the three-dimensional feature map $F \triangleq \phi(x; W^*)$ produced by the DSVDD to see how a small variation in it can affect the final likelihood output of the Classifier y:

$$\alpha_k = \frac{1}{Z} \sum_i \sum_j \frac{\partial y'}{\partial F_{ij}^k}$$

where Z is a constant to lead to the global average, y' is the output of the Classifier before the *sigmoid* activation, and thus α_k implies the average impact that the features in the kth channel make to increase y' as well as y. Then, the final Grad-CAM heatmap at i, j can easily be computed as follows:

$$h_{ij} = ReLU(\sum_k \alpha_k F_{ij}^k)$$

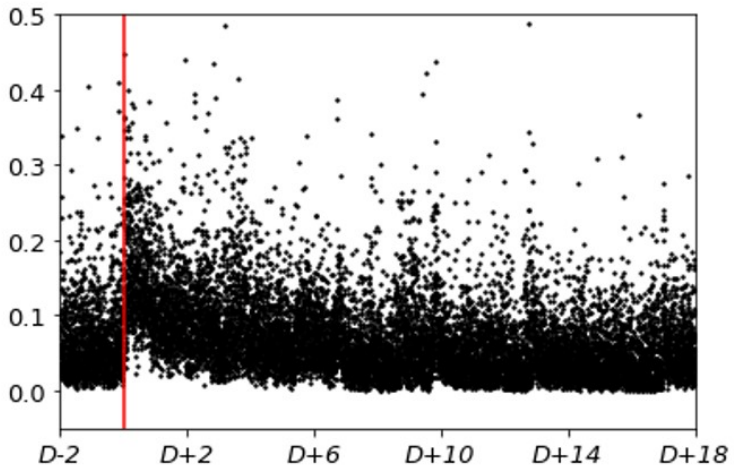


Figure 4.7: Optical flow weights each sampled at the interval of over 2 minutes for 20 days. The horizontal axis shows the days of observation with the red separator when the *gamergates* were intentionally moved. The vertical axis indicates the weight levels, each standardized in $[0, 1]$ by the global max and min, omitting extreme outliers for clarity.

where $ReLU$ works as the identity function for the the positive inputs, but it passes 0 for the negative (i.e., it is a rectifier). It is used to consider only the features that bring positive impacts on the likelihoods of *unstable* state – when increased, y also raises, and also, the entire heatmap H can be resized to superimpose on the original image. In Section 4.6.5, we will discuss the obtained results on the real ant images.

4.7 Experiments

We are currently ready to examine the effectiveness of the proposed prediction framework with well designed experiments. Firstly, we will try to manually find any patterns from the extracted optical flow datasets to discuss whether sophisticated predictive models are needed for the OC task in the focal system. Then, the IO-GEN based approach will be tested primarily with comparisons with other baselines,

especially the ones that could justify the structural designs of the IO-GEN pipeline. Interesting internal behaviors of the IO-GEN model itself will also be analyzed to explain how the successful performance could be achieved. Lastly, Grad-CAM is adopted to show its potential use case to reveal behavioral features from natural, living agents in a human-understandable way.

4.7.1 Analysis of Optical Flow Weights

We first attempt to make use of the optical flow dataset to discover insightful motional patterns without any complex model. In particular, as with Mahasseni *et al.* (2013), we compute an optical flow weight w_i for each frame i by averaging the magnitudes of flow vectors at all locations. Fig. 4.7 displays the obtained weight signal in time as $m = 1$ optical flow frame is considered for each sampling interval.

For first two days, the weights generally stay in a certain range implying some behavioral regularity maintained among ants. Yet, an obvious increase is noticed immediately when the gamergates are moved away starting D+1, because the negative event triggered a social tournament in which frequent aggresive behaviors were elicited. As the unstable state develops, the magnitude level continues to decrease and finally recovers the original extent roughly on D+10, even though we could still find several ants that presented hostile interactions.

Consequently, a simple model could be attempted to use the overall rise of flow weight as the only feature to distinguish unstable ants from stable ones especially at early development of unstable state. Following experimental results, however, will provide concrete examples to explain the limitations of such design and the need of more complex models for reliable predictions.

m	1	2	4
AUC	0.760 ± 0.016	0.786 ± 0.009	0.787 ± 0.008

Table 4.1: Average performance when the number of optical flow image frames per input is set to $1, 2$, or 4.

4.7.2 Evaluations

We here demonstrate the OC performance of our proposed method, first with an ablation study to find the best number of optical flow frames per input. Next, the baselines used for comparison are introduced, and ours competes with them to explore its overall reliability as well as robust prediction in various time windows during colonial stabilization.

Following protocols of previous works (Ruff *et al.*, 2018), the Area Under the Curve (AUC) of the Receiver Operating Characteristics (ROC) are measured for each model to reflect the separability between classes. Particularly, the average over three distinct splits is reported with the standard deviation when needed.

4.7.2.1 Effects of Observational Length

From the test with $m \in \{1, 2, 4\}$, Table 4.1 reports that there was an improvement as m increased from 1 to 2, while doubling it to 4 did not offer any benefit. The result may indicate that the observation of one more second does not add significantly more information. Learning IO-GEN could also be more challenging as it is asked to generate longer motional sequences. Thus, m is set to 2 hereafter considering both efficiency and effectiveness of our model.

4.7.2.2 Baseline Models

OFW uses the temporal optical flow weights to set the best threshold to report the best classification result. **DCAE** is a similar threshold-based method relying on the reconstruction error as the feature of novelty (Kerner *et al.*, 2019). **OC-SVM** (Schölkopf *et al.*, 2001) takes the encoder of DCAE to build the One-class SVM on it providing the performance with the best ν parameter. While **DSVDD** here is designed similarly to the description by Ruff *et al.* (2018), the adjustments in our implementation are described in Section 4.6.4 above. **GEN** and **N-GEN** are generative models to train a separate classifier as our method. GEN is, however, a standard generative model adopting the feature matching technique in the discriminator network instead without the intervention of DSVDD. N-GEN replaces $\phi(G(z))$ with arbitrary noisy data $\vec{v}' \in \mathbb{R}^{1\times 2048}$ where each element of $\vec{v}'$ is drawn from $N(0, \alpha)$ where α is the global variation of $\vec{v} \sim \phi(G(z))$.

4.7.2.3 Overall Detection Performance

Table 4.2 helps estimate overall reliability of each model for the image inputs that can be captured at an arbitrary timing since all samples from unstable colony were included for test. OFW and DCAE suggest the limitation of only relying on thresholding a simplistic one-dimensional signal. In particular, the low accuracy of DCAE implies that precise reconstruction is achieved also for unseen, unstable motions. Similarly, the OC-SVM can utilize only little benefit from the encoding capability. On the other hand, DSVDD leads at least 45% increase of AUC score simply fine-tuning the encoder part of DCAE because unstable examples are more easily distinguished in the newly learned hyperspheric data description. In addition, our model brings about a further improvement proving that utilizing a subsequent classifier with syn-

METHOD	AUC
OFW	0.506
DCAE	0.506 $\pm$ 0.002
OC-SVM	0.523 $\pm$ 0.004
DSVDD	0.762 $\pm$ 0.013
GEN	0.587 $\pm$ 0.032
N-GEN	0.699 $\pm$ 0.006
IO-GEN	**0.786 $\pm$ 0.009**

Table 4.2: Average AUC of tested models with the standard deviation as all 18-day unstable observations are considered.

thetic examples can be more effective than the distancing heuristic in DSVDD to make full use of multi-dimensional relationships among features. Nevertheless, GEN and N-GEN provide 25% and 16% poorer performance than ours although both also use synthetic data to train a classifier. N-GEN actually performs better than GEN implying that the prior knowledge on data description is useful for effective data synthesis. Still, its insufficient reliability emphasizes the realism in generated datasets as well.

4.7.2.4 Results in Different Developmental Phases

Figure 4.8 displays the performance variation of each model as the tested unstable ant data are confined in various temporal windows. Consistent with Fig. 4.7, the prediction performance generally degrades for later temporal bins because ant behaviors are more stabilized. Our framework still demonstrates the top performance in almost

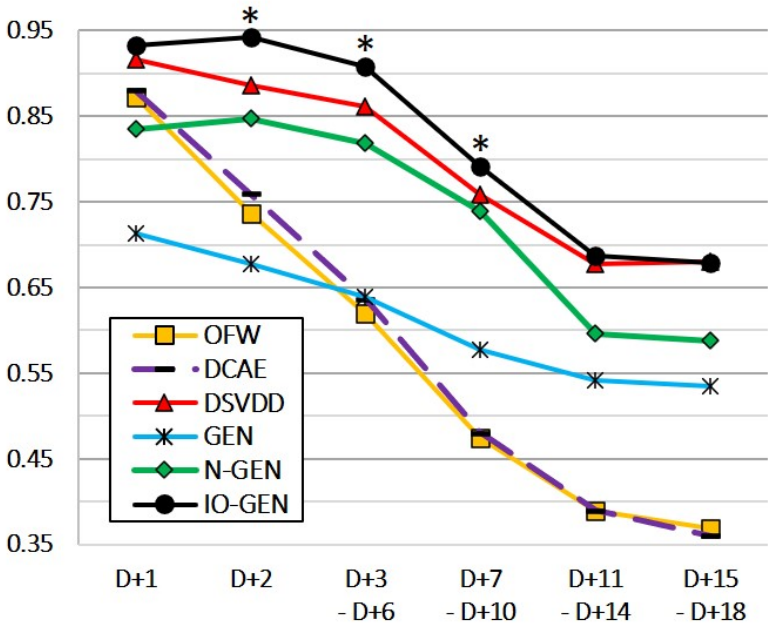


Figure 4.8: Average AUC changes for predictions within different temporal windows. * marks statistically significant improvement against DSVDD ($p < .05$).

any phase especially showing the highest margins from DSVDD while the colony could induce most diverse motions with the dramatic stabilization between D+2 and D+10. As expected from Fig. 4.7, OFW and DCAE highly depend on the timing of application because their scores are close to that of DSVDD early while lower even than 0.5 after D+6. If the initial social transition is weaker, these models may perform poorly due to less intense competition caused. As in the previous experiment, the results from GEN and N-GEN highlight the importance of using knowledge of feature space to generate data as a proxy of unseen class. As illustrated in Fig. 4.2, GEN produces fake motion samples that closely resemble stable ants, and so the classifier cannot perform well even on abnormal samples in the early days just after the social transition is artificially triggered.

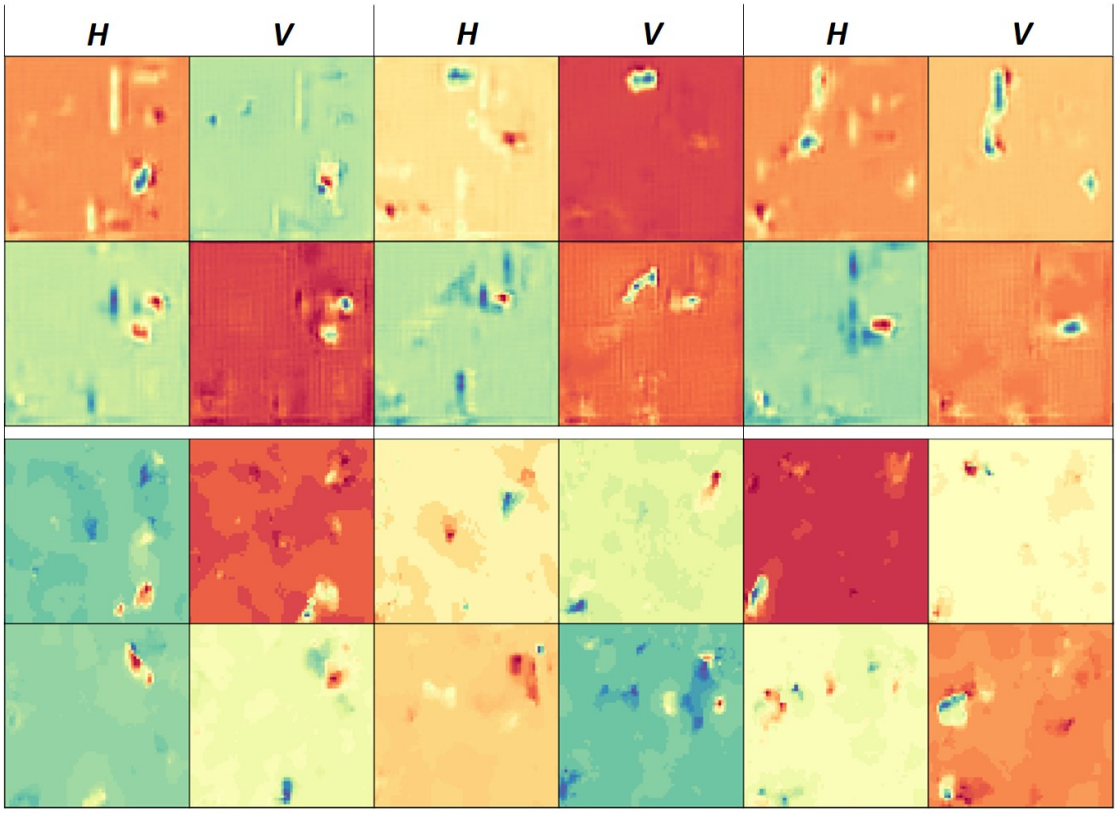


Figure 4.9: Optical flow examples: (top two rows) Six synthesized pairs from IO-GEN; (bottom two rows) Six real examples. Each (H-V) pair show horizontal and vertical motions, respectively, for which pixels are normalized in each image in a colormap.

4.7.3 Model Properties

Figure 4.9 compares synthetic optical flows from IO-GEN to real optical flows; the generated optical flows are visually similar to real flows. Furthermore, Fig. 4.10a illustrates that the lowest distance distribution to $\vec{c}$ is measured with IO-GEN, as designed, whereas GEN behaves similarly to the stable dataset. Fig. 4.10b finally shows the predictive outcomes of Classifier, which are likelihoods of unstable state. With the *label switch*, the confidence becomes positively correlated with the distance to $\vec{c}$ viewing inner outliers as samples from the most stable colony. Clear differences

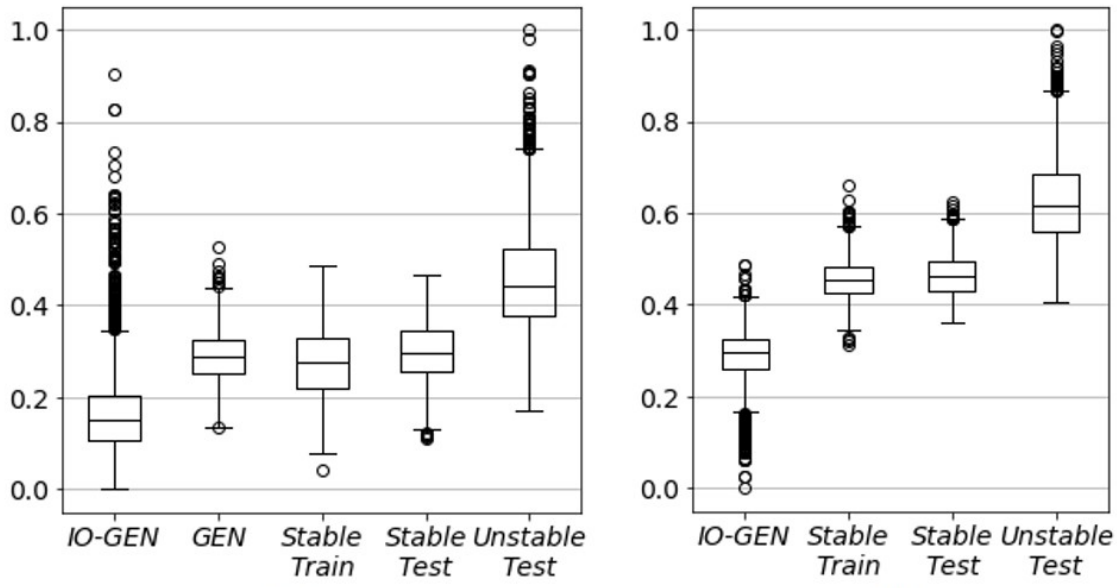


Figure 4.10: For different types of data: On the left, normalized Euclidean distances to $\vec{c}$ in feature description $\mathcal{F}$ of DSVDD. On the right, predicted likelihoods from Classifier.

between classes imply that learned knowledge to discriminate stable and more-stable states in DSVDD can be transferred for classification of another pair as stable or unstable.

4.7.4 Identification of Salient Interactions

So far, we have investigated a new method to precisely classify the binary social state only when behavioral observations of only one state are available to model the world. The approach has shown its success though the focal system inherits the realistic challenge **C4** and also does not allow motional state search strategies, such as *SRS* in Chapter 3, to be directly applicable. In this section, we will explore the feasible use of Grad-CAM in the proposed state estimation pipeline, motivated by the fifth challenge, **C5**, in Section 1.3.

Figure 4.11: Two examples in two rows in each of which Grad-CAM heatmap is resized and superimposed on original RGB video frames sequentially sampled every second. Red regions have higher values while the blue lower values.

For evaluation, the Grad-CAM module is set up on the (DSVDD, Classifier) estimator, and higher values in the output heatmap represent the positive influence on the likelihood of stable state. Thus, we will manually verify whether the *dueling* interactions are captured on input frames during the unstable period. Because the strong association between unstable colony and *dueling* is already known, it is expected that the containing regions tend to have high intensities in the heatmap. Moreover, from the results, we could estimate the potential ability of the model to discover crucial behavioral cues for macrostate estimation in other types of society.

Figure 4.11 offers two examples from the *naïve* Grad-CAM method where the heatmaps are superimposed on the original RGB video frames instead of the input optical flows for better scenic understanding. Obviously, there is very little difference in the heatmaps between time stamps within the same example and also across examples, even though the central areas commonly show higher gradients than near

boundaries implying that the Classifier hypothesizes relatively strong associations of any behaviors at the broad center area with the social instability in any case. This can be a wisely learned strategy to consider the observed trend from Fig. 4.7, with which unstable periods generally involve more intensive behaviors with higher optical flow magnitudes especially at early stages.

To more clearly investigate the effects of microlevel interactions, the heatmap algorithm is configured to 1) subtract the position-wise global minimum from the heatmap value at each x,y position and 2) only display top 5% regions. Fig. 4.12 and Fig. 4.13 provide 6 examples in which *dueling* interactions occurred and were properly recognized by the Classifier to predict unstable colonies. In fact, this capability supports that our model with Grad-CAM could be used to reveal cryptic behavioral cues also in other biological systems.

4.8 Concluding Remarks

As stated in the beginning of this chapter, we have investigated an alternative approach to the method in Chapter 3 to estimate *social stability* in ant colonies that inherit the fourth challenge (**C4**) introduced in Section 1.3. Our approach allows for some *bias* in the training dataset in which behavioral samples only from the stable state are contained albeit the ultimate goal is to also detect the unstable state. With the *monotonic* data, the detection problem has been re-framed in the aspect of OC to make full use of the biasness.

We have introduced a novel generative model IO-GEN, which can take advantage of pre-trained DSVDD and a separate classifier to successfully solve the problem of OC. Our framework has been applied to a 20-day video recording from a nest of over 50 ants of *H. saltator* to identify a colony's stable or unstable state only from a 1-second motional sequence. Experimental results have demonstrated that the classifier

trained with the synthetic data from IO-GEN can outperform other state-of-the-art baselines at any temporal phase while the insect society is stabilized.

Even further, an additional module based on Grad-CAM has demonstrated its potential use to tackle **C5**. The results enabled to ensure that the estimator can not only learn reasonable holistic patterns that are consistent with the prior observations on behavioral trends in optical flows but also discover the individual motions that might have been cryptic to human observers.

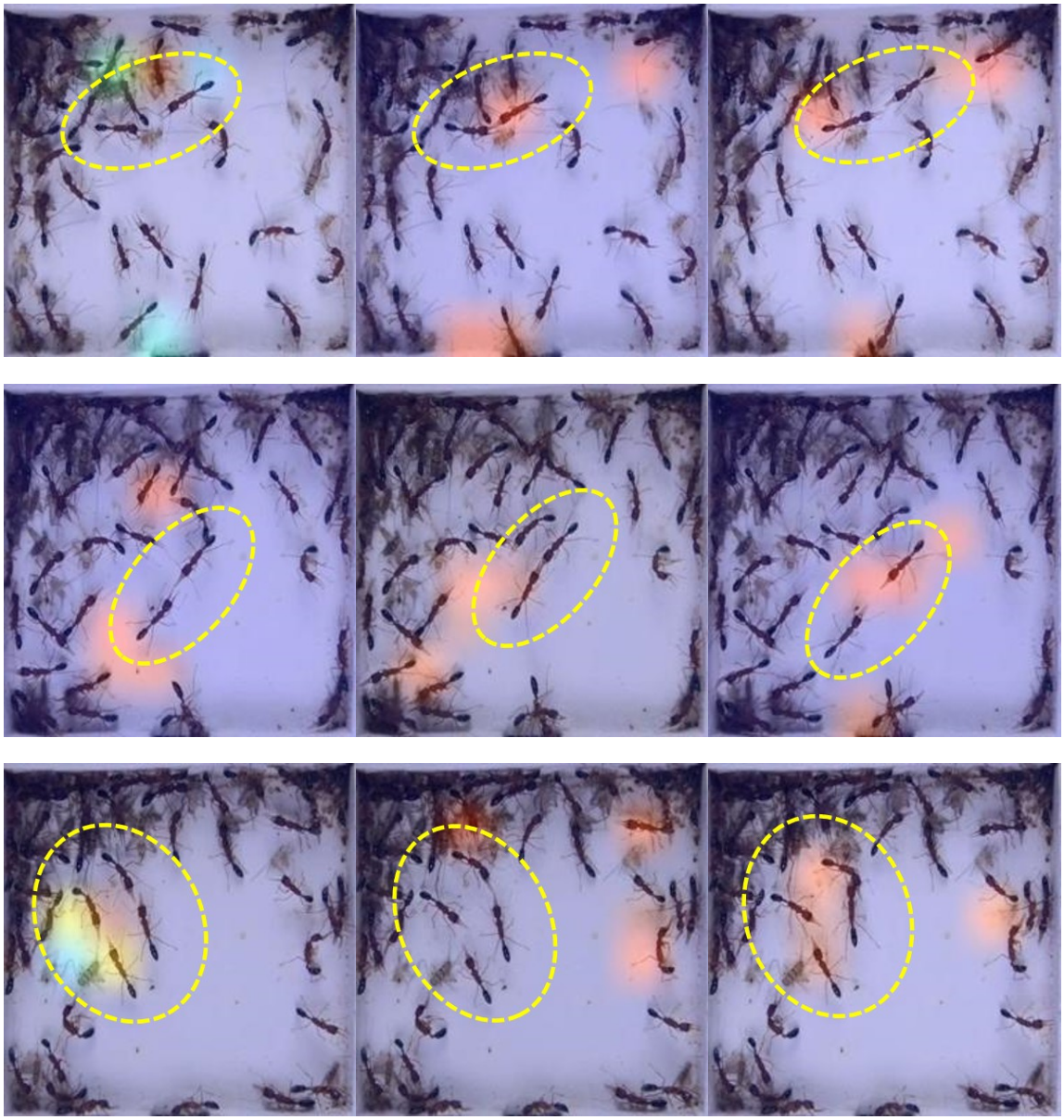


Figure 4.12: Three *dueling* examples in three rows as only the regions of top 5% positive gradients are visualized for each frame. The Classifier focuses around the duelers (manually located with dashed ovals) and a few of other movers.

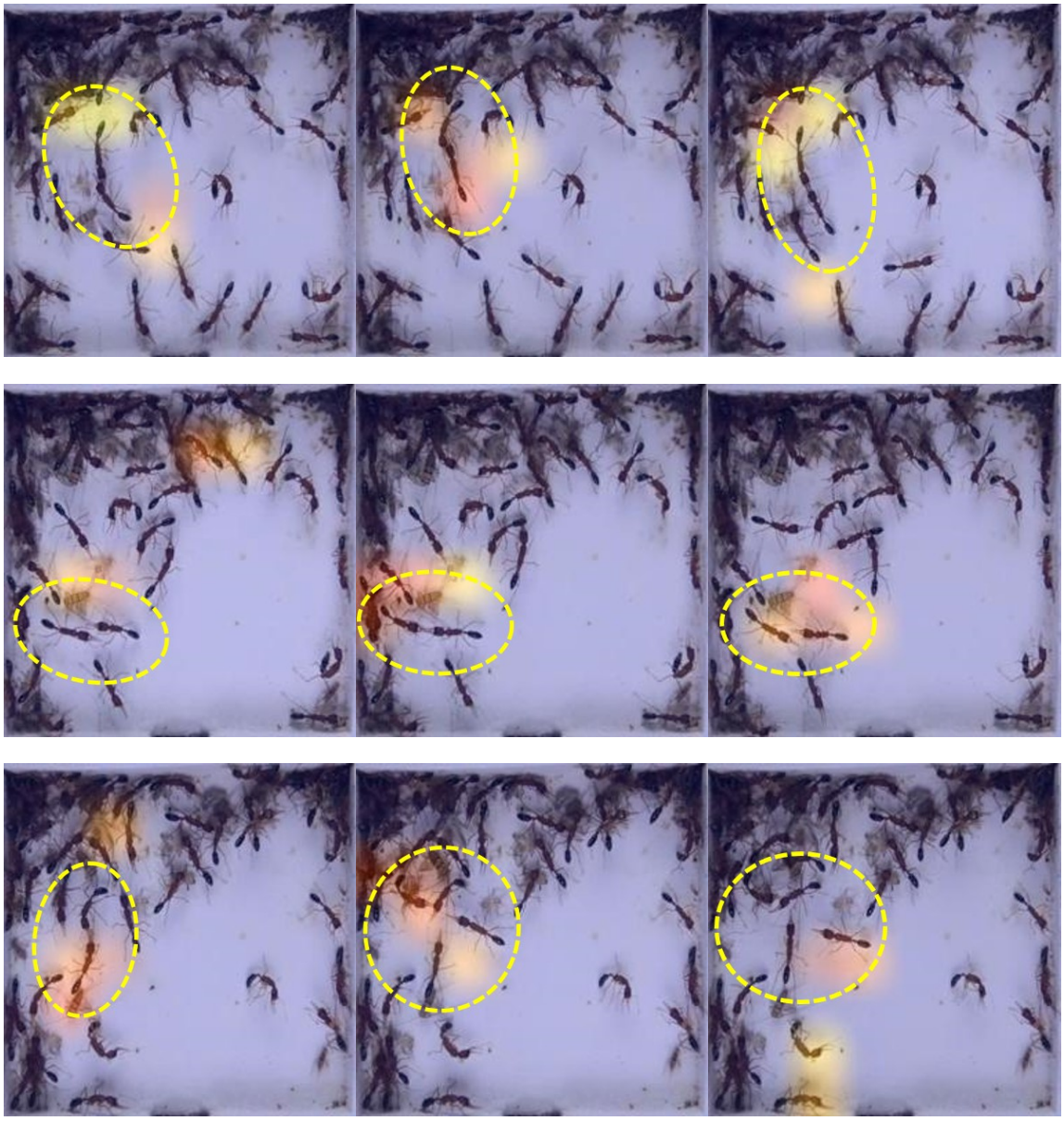


Figure 4.13: Other three *dueling* examples in three rows. Similar properties to Fig. 4.12 are observed.

Chapter 5

CONCLUSION

In this chapter, we will conclude the dissertation summarizing all the contributions and discussing the potential directions of future work.

5.1 Summary of Contributions

The main contribution of this dissertation is, first of all, to formalize *macrostate estimation* problem in complex multi-agent systems, motivated by the historical attempts of viewing a complex social system as a single, large agent to abstract all microlevel events. The advantages of being able to infer macrostates have also been described with *five challenges* **C1∼C5** that could make it difficult to perform the estimation in any realistic systems. Successful approaches have been introduced to address each challenge across artificial and natural multi-agent systems. Specific contributions are described below.

5.1.1 Remote Teammate Localization [1]

Chapter 2 formalized the ReTLo problem in multi-robot teams in which an individual robot is trained to predict positional states of the whole system only from local views of a nearby neighbor. I introduced a "modular" estimation approach to effectively tackle **C1∼C3**, and its reliability was evaluated on both simulated and physical robotic platforms. The results have supported the utility of the approach for the robotic members to successfully coordinate themselves under practical, unfavorable conditions where remote communication is not allowed, or proximity sensors

[1] (Choi *et al.*, 2017), (Choi *et al.*, 2020)

produce noisy measurements. A novel strategy was also proposed to achieve caging behavior using the ReTLo ability. Furthermore, the real-robot datasets have been uploaded online to encourage more participations from the research community.

5.1.2 Active Actions for Non-biased Learning [2]

In Chapter 3, an improved learning model of ReTLo was introduced in which the individual robotic learner is designed to have a capability of moving "actively" to access "non-biased" observations during the stage of data collection. This method was invented to mitigate **C4** by allowing the learner to select its actions independently of the pre-programmed motion rule so as to avoid gathering only similar, biased observations over time. To realize effective activeness, I suggested random sampling of actions incorporated with original designs of neural networks that are trained in unsupervised and self-supervised manners simultaneously to predict novelties of reachable states and only select the optical action. The experimental results demonstrated that the active data sampling can lead a high performance learning with less but better representative data while the identical estimator is used with the previous passive model.

5.1.3 Intentional Biased Learning for State Detection in Insect Colony [3]

In Chapter 4, social stability was estimated in an ant colony, which is a highly complex social system accompanying **C1**. As opposed to Chapter 3, I demonstrated that learning with "biased" datasets can still be useful to estimate macrostates under **C4** without controlling particular micro-agents. Motivated by one-class classification approaches, I proposed a framework to distinguish "unstable" social states when only

[2](Choi and Pavlic, 2020)

[3](Choi *et al.*, 2021)

the data of "stable" states are available for training. In particular, I have shared online the used video data of 20-day motional observations from over 50 ants while their colony transitioned through *stable–unstable–stable* states. To the best of my knowledge, I showed the first work to learn a generative model that synthesizes fake behavioral representations of ant, and also, the artificial dataset could assist in learning a high-performance classifier for the OC task. Program codes are also available online. Moreover, the Grad-CAM technique was adopted as a promising solution to solve **C5** visually highlighting "salient" micro-behaviors which may have been enigmatic to human observers in the relationship with macroscale states.

5.2 Future Directions

In the future work of ReTLo, we could more deeply investigate the factors in model accuracy such as the length of history or types of team behavior that might favor the prediction scheme. Specifically, Reinforcement Learning based methods could be devised for the *Tail* robot to stay active even after training to maximize the localization performance – i.e., once a trained estimator has been equipped on *Tail*, it could try to learn its own motion rules to improve its remote localization ability by continuously interacting with the environments. Then, the learned motions could drive the team to stay within a certain range of formations on which the localization error can be minimized in average, however *Head* leads the team at a distal end.

Also, because the LSTM layer is exposed to various evolutions of team shape during training, the vector representation of it may be examined to characterize team states and ultimately detect large-scale group-shape abnormalities that may be important to detect for situational awareness. Beyond these approaches, Bayesian implementations could be considered to make better use of uncertainty information from sensors and provide confidence estimates for the position of each robot in the

team.

In addition, we could explore applicable cases of ReTLo beyond caging scenario where multiple *Tail* robots are deployed in the same team to cooperatively drive other robots to a destination state based on their real-time predictions of global formation. Moreover, an autonomous vehicle could utilize a ReTLo-like inference model to predict potential accidents that may have occurred several cars ahead using the behavioral variations of the car immediately in front.

With the natural systems, one of the future directions could be to quantify the macrostates at a higher resolution. For instance, a colony of *Harpegnathos saltator* undergoes a stabilization process taking multiple days or weeks until an equilibrium of stability is reached. This means that within a small time window of minutes or hours, the colony may experience different stages of states although behavioral changes could look very subtle to human eyes. If a model can be learned to detect such a unnoticeable progression, it could help human observers better assess the system dynamics and also learn about more diverse salient behaviors using the Grad-CAM technique.

Antonelli, G., F. Arrichiello, F. Caccavale and A. Marino, "A decentralized controller-observer scheme for multi-agent weighted centroid tracking", IEEE Transactions on Automatic Control **58**, 5, 1310–1316 (2012).

Arkin, R. C., *Behavior-Based Robotics* (MIT Press, 1998).

Berger, M., L. M. Seversky and D. S. Brown, "Classifying swarm behavior via compressive subspace learning", in "2016 IEEE International Conference on Robotics and Automation (ICRA)", pp. 5328–5335 (IEEE, 2016).

Bishop, C. M., "Pattern recognition", Machine Learning **128**, 1–58 (2006).

Bozek, K., L. Hebert, A. S. Mikheyev and G. J. Stephens, "Towards dense object tracking in a 2d honeybee hive", in "Proc. IEEE CVPR 2018", pp. 4185–4193 (2018).

Bradbury, J. W. and S. L. Vehrencamp, *Principles of Animal Communication* (Sinauer Associates, Inc., 2011), second edn.

Brambilla, M., E. Ferrante, M. Birattari and M. Dorigo, "Swarm robotics: a review from the swarm engineering perspective", Swarm Intelligence **7**, 1, 1–41 (2013).

Brown, D. S. and M. A. Goodrich, "Limited bandwidth recognition of collective behaviors in bio-inspired swarms", in "Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems", pp. 405–412 (International Foundation for Autonomous Agents and Multiagent Systems, 2014).

Buhl, J., D. J. Sumpter, I. D. Couzin, J. J. Hale, E. Despland, E. R. Miller and S. J. Simpson, "From disorder to order in marching locusts", Science **312**, 5778, 1402–1406 (2006).

Burgard, W., D. Fox and S. Thrun, "Active mobile robot localization", in "IJCAI", pp. 1346–1352 (1997).

Byravan, A. and D. Fox, "Se3-nets: Learning rigid body motion using deep neural networks", in "2017 IEEE International Conference on Robotics and Automation (ICRA)", pp. 173–180 (IEEE, 2017).

Cakmak, M. and A. L. Thomaz, "Designing robot learners that ask good questions", in "2012 7th ACM/IEEE Intl. Conf. on Human-Robot Interaction (HRI)", pp. 17–24 (2012).

Caley, J. A. and G. A. Hollinger, "Environment prediction from sparse samples for robotic information gathering", in "2020 IEEE Intl. Conf. on Robotics and Automation (ICRA)", (2020).

Calhoun, A. J., J. W. Pillow and M. Murthy, "Unsupervised identification of the internal states that shape natural behavior", Nature Neuroscience **22**, 12, 2040–2049 (2019).

Chao, C., M. Cakmak and A. L. Thomaz, "Transparent active learning for robots", in "2010 5th ACM/IEEE Intl. Conf. on Human-Robot Interaction (HRI)", pp. 317–324 (2010).

Chaplot, D. S., E. Parisotto and R. Salakhutdinov, "Active neural localization", arXiv preprint arXiv:1801.08214 (2018).

Chen, N., A. Klushyn, A. Paraschos, D. Benbouzid and P. Van der Smagt, "Active learning based on data uncertainty and model sensitivity", in "2018 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)", pp. 1547–1554 (2018).

Cheng, Y. and D. Xie, "Distributed observer design for bounded tracking control of leader–follower multi-agent systems in a sampled-data setting", International Journal of Control **87**, 1, 41–51 (2014).

Choi, T., S. Kang and T. P. Pavlic, "Learning local behavioral sequences to better infer non-local properties in real multi-robot systems", in "2020 IEEE Intl. Conf. on Robotics and Automation (ICRA)", (2020).

Choi, T. and T. P. Pavlic, "Automatic discovery of motion patterns that improve learning rate in communication-limited multi-robot systems", in "2020 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)", pp. 243–248 (IEEE, 2020).

Choi, T., T. P. Pavlic and A. W. Richa, "Automated synthesis of scalable algorithms for inferring non-local properties to assist in multi-robot teaming", in "2017 13th IEEE Conference on Automation Science and Engineering (CASE)", pp. 1522–1527 (IEEE, 2017).

Choi, T., B. Pyenson, J. Liebig and T. P. Pavlic, "Identification of abnormal states in videos of ants undergoing social phase change", in "Proceedings of the AAAI Conference on Artificial Intelligence", (2021).

Cohn, D., L. Atlas and R. Ladner, "Improving generalization with active learning", Machine learning **15**, 2, 201–221 (1994).

Correll, N. and A. Martinoli, "Modeling and optimization of a swarm-intelligent inspection system", in "Proc. of the Seventh International Symposium on Distributed Autonomous Robotics Systems (DARS 2004)", pp. 369–378 (Toulouse, France, 2004).

Couzin, I., "Collective minds", Nature **445**, 7129, 715–715 (2007).

Couzin, I. D., J. Krause, R. James, G. D. Ruxton and N. R. Franks, "Collective memory and spatial sorting in animal groups", J. Theor. Biol. **218**, 1, 1–12 (2002).

Das, B., M. S. Couceiro and P. A. Vargas, "MRoCS: A new multi-robot communication system based on passive action recognition", Robotics and Autonomous Systems **82**, 46–60 (2016).

De Silva, O., G. K. I. Mann and R. G. Gosine, "Efficient distributed multi-robot localization: A target tracking inspired design", in "Proc. of the 2015 IEEE International Conference on Robotics and Automation", pp. 434–439 (2015).

Dekel, O., C. Gentile and K. Sridharan, "Selective sampling and active learning from single and multiple teachers", Journal of Machine Learning Research **13**, Sep, 2655–2697 (2012).

Derakhshandeh, Z., R. Gmyr, A. W. Richa, C. Scheideler, T. Strothmann and S. Tzur-David, "Infinite object coating in the amoebot model", arXiv preprint arXiv:1411.2356 (2014).

Elamvazhuthi, K. and S. Berman, "Scalable formation control of multi-robot chain networks using a PDE abstraction", in "Proc. of the 12th International Symposium on Distributed Autonomous Robotic Systems", pp. 357–369 (2016).

Emerson, A. E., "Social coordination and the superorganism", American Midland Naturalist **21**, 1, 182–209 (1939).

Fox, D., W. Burgard and S. Thrun, "Active markov localization for mobile robots", Robotics and Autonomous Systems **25**, 3-4, 195–207 (1998).

Franchi, A., P. Stegagno, M. Di Rocco and G. Oriolo, "Distributed target localization and encirclement with a multi-robot system", in "Proc. of the 7th IFAC Symposium on Intelligent Autonomous Vehicles", pp. 151–156 (2010).

Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville and Y. Bengio, "Generative adversarial nets", in "Proc. NIPS 2014", pp. 2672–2680 (2014).

Gottipati, S. K., K. Seo, D. Bhatt, V. Mai, K. Murthy and L. Paull, "Deep active localization", IEEE Robotics and Automation Letters **4**, 4, 4394–4401 (2019).

Grocholsky, B., V. Kumar and H. Durrant-Whyte, "Anonymous cooperation in robotic sensor networks", in "Proc. of the AAAI-04 Workshop on Sensor Networks", (2004).

Hara, K., R. Vemulapalli and R. Chellappa, "Designing deep convolutional neural networks for continuous object orientation estimation", arXiv preprint arXiv:1702.01499 (2017).

Heinze, J., B. Hölldobler and C. Peeters, "Conflict and cooperation in ant societies", Naturwissenschaften **81**, 11, 489–497 (1994).

Hobbes, T., *Leviathan: Or, The Matter, Forme and Power of Commonwealth, Ecclesiasticall and Civill* (University Press, 1904).

Hochreiter, S. and J. Schmidhuber, "Long short-term memory", Neural computation **9**, 8, 1735–1780 (1997).

Kerner, H. R., D. F. Wellington, K. L. Wagstaff, J. F. Bell, C. Kwan and H. B. Amor, "Novelty detection for multispectral images with application to planetary exploration", in "Proc. AAAI Conf. on Artificial Intelligence", vol. 33, pp. 9484–9491 (2019).

Liebig, J., C. Peeters and B. Hölldobler, "Worker policing limits the number of reproductives in a ponerine ant", Proc. R. Soc. B **266**, 1431, 1865–1870 (1999).

Luft, L., T. Schubert, S. I. Roumeliotis and W. Burgard, "Recursive decentralized collaborative localization for sparsely communicating robots.", in "Robotics: Science and Systems", (New York, NY, USA, 2016).

Maeda, G., M. Ewerton, T. Osa, B. Busch and J. Peters, "Active incremental learning of robot movement primitives", (2017).

Mahasseni, B., S. Chen, A. Fern and S. Todorovic, "Detecting the moment of snap in real-world football videos.", in "Proc. IAAI 2013", (2013).

Mehran, R., A. Oyama and M. Shah, "Abnormal crowd behavior detection using social force model", in "Proc. IEEE CVPR 2009", pp. 935–942 (2009).

Michel, O., "Webots: Professional mobile robot simulation", Journal of Advanced Robotics Systems **1**, 1, 39–42 (2004).

Napp, N. and E. Klavins, "A compositional framework for programming stochastically interacting robots", International Journal of Robotics Research **30**, 6, 713–729 (2011).

Nath, T., A. Mathis, A. C. Chen, A. Patel, M. Bethge and M. W. Mathis, "Using deeplabcut for 3d markerless pose estimation across species and behaviors", Nature Protocols **14**, 7, 2152–2176 (2019).

Novitzky, M., C. Pippin, T. R. Collins, T. R. Balch and M. E. West, "Bio-inspired multi-robot communication through behavior recognition", in "Proc. of the 2012 IEEE Conference on Robotics and Biomimetics (ROBIO)", pp. 771–776 (2012).

Odhner, L. U. and H. Asada, "Stochastic recruitment control of large ensemble systems with limited feedback", Journal of Dynamic Systems, Measurement, and Control **132**, 4 (2010).

Oh, D. Y. and I. D. Yun, "Residual error based anomaly detection using auto-encoder in smd machine sound", Sensors **18**, 5, 1308 (2018).

Palo, N. D. and E. Johns, "Active robot imitation learning with autoencodersand imagined rollouts", URL `http://www.robot-learning.ml/2019/` (2019).

Peeters, C. and B. Hölldobler, "Reproductive cooperation between queens and their mated workers: the complex life history of an ant with a valuable nest", Proc. Natl. Acad. Sci. USA **92**, 24, 10977–10979 (1995).

Perera, P., R. Nallapati and B. Xiang, "Ocgan: One-class novelty detection using gans with constrained latent representations", in "Proc. IEEE CVPR 2019", pp. 2898–2906 (2019).

Pickem, D., M. Lee and M. Egerstedt, "The GRITSBot in its natural habitat – a multi-robot testbed", in "Proc. of the 2015 IEEE International Conference on Robotics and Automation", pp. 4062–4067 (2015).

Pickem, D., L. Wang, P. Glotfelter, Y. Diaz-Mercado, M. Mote, A. Ames, E. Feron and M. Egerstedt, "Safe, remote-access swarm robotics research on the robotarium", arXiv preprint arXiv:1604.00640 (2016).

Poole, D. L. and A. K. Mackworth, *Artificial Intelligence: foundations of computational agents* (Cambridge University Press, 2010).

Pratt, S. C., E. B. Mallon, D. J. Sumpter and N. R. Franks, "Quorum sensing, recruitment, and collective decision-making during colony emigration by the ant *leptothorax albipennis*", Behav. Ecol. Sociobiol. **52**, 2, 117–127 (2002).

Reid, C. R., M. J. Lutz, S. Powell, A. B. Kao, I. D. Couzin and S. Garnier, "Army ants dynamically adjust living bridges in response to a cost–benefit trade-off", Proc. Natl. Acad. Sci. USA **112**, 49, 15113–15118 (2015).

Ribeiro, M., M. Gutoski, A. E. Lazzaretti and H. S. Lopes, "One-class classification in images and videos using a convolutional autoencoder with compact embedding", IEEE Access **8**, 86520–86535 (2020).

Richter, C. and N. Roy, "Safe visual navigation via deep learning and novelty detection", (2017).

Riedo, F., M. Chevalier, S. Magnenat and F. Mondada, "Thymio II, a robot that grows wiser with children", in "2013 IEEE Workshop on Advanced Robotics and its Social Impacts", (Tokyo, Japan, 2013).

Rubenstein, M., A. Cornejo and R. Nagpal, "Programmable self-assembly in a thousand-robot swarm", Science **345**, 6198, 795–799 (2014).

Ruff, L., R. A. Vandermeulen, N. Görnitz, L. Deecke, S. A. Siddiqui, A. Binder, E. Müller and M. Kloft, "Deep one-class classification", in "Proc. ICML 2018", vol. 10, pp. 6981–6996 (2018).

Sabokrou, M., M. Khalooei, M. Fathy and E. Adeli, "Adversarially learned one-class classifier for novelty detection", in "Proc. IEEE CVPR 2018", pp. 3379–3388 (2018).

Salimans, T., I. Goodfellow, W. Zaremba, V. Cheung, A. Radford and X. Chen, "Improved techniques for training gans", in "Proc. NIPS 2016", pp. 2234–2242 (2016).

Sasaki, T., C. A. Penick, Z. Shaffer, K. L. Haight, S. C. Pratt and J. Liebig, "A simple behavioral model predicts the emergence of complex animal hierarchies", The American Naturalist **187**, 6, 765–775 (2016).

Schmidhuber, J., "Deep learning in neural networks: An overview", Neural Networks **61** (2015).

Schölkopf, B., J. C. Platt, J. Shawe-Taylor, A. J. Smola and R. C. Williamson, "Estimating the support of a high-dimensional distribution", Neural Computation **13**, 7, 1443–1471 (2001).

Selvaraju, R. R., M. Cogswell, A. Das, R. Vedantam, D. Parikh and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization", in "Proceedings of the IEEE international conference on computer vision", pp. 618–626 (2017).

Settles, B., "Active learning literature survey", Tech. rep., University of Wisconsin-Madison Department of Computer Sciences (2009).

Simonyan, K. and A. Zisserman, "Two-stream convolutional networks for action recognition in videos", in "Proc. NIPS 2014", pp. 568–576 (2014).

Tax, D. M. and R. P. Duin, "Support vector data description", Machine Learning **54**, 1, 45–66 (2004).

Upton, E. and G. Halfacree, *Raspberry Pi user guide* (John Wiley & Sons, 2014).

Valentini, G., E. Ferrante, H. Hamann and M. Dorigo, "Collective decision with 100 Kilobots: speed versus accuracy in binary discrimination problems", Autonomous Agents and Multi-Agent Systems **30**, 3, 553–580 (2016).

Valentini, G. and H. Hamann, "Time-variant feedback processes in colletive decision-making systems: influence and effect of dynamic neighborhood sizes", Swarm Intelligence **9**, 2, 153–176 (2015).

Valentini, G., H. Hamann and M. Dorigo, "Efficient decision-making in a self-organizing robot swarm: On the speed versus accuracy trade-off", in "Proc. of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015)", pp. 1305–1314 (Istanbul, Turkey, 2015).

Von Frisch, K., *The Dance Language and Orientation of Bees* (Harvard University Press, 1967).

Wang, L., Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang and L. Van Gool, "Temporal segment networks: Towards good practices for deep action recognition", in "European conference on computer vision", pp. 20–36 (2016).

Wang, Z. and V. Kumar, "Object closure and manipulation by multiple cooperating mobile robots", in "Proc. of the 2002 IEEE International Conference on Robotics and Automation", pp. 394–399 (2002).

Wilson, S., T. P. Pavlic, G. P. Kumar, A. Buffin, S. C. Pratt and S. Berman, "Design of ant-inspired stochastic control policies for collective transport by robotic swarms", Swarm Intelligence **8**, 4, 303–327 (2014).

Wu, Y., M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation", arXiv preprint arXiv:1609.08144 (2016).

Xiao-Yuan, L., H. Na-Ni and G. Xin-Ping, "Leader-following formation control of multi-agent networks based on distributed observers", Chinese Physics B **19**, 10 (2010).

Xiong, Y. and R. Zuo, "Recognition of geochemical anomalies using a deep autoencoder network", Computers & Geosciences **86**, 75–82 (2016).

Xu, D., E. Ricci, Y. Yan, J. Song and N. Sebe, "Learning deep representations of appearance and motion for anomalous event detection", arXiv preprint arXiv:1510.01553 (2015).

Yadav, S., C. Chen and A. Ross, "Relativistic discriminator: A one-class classifier for generalized iris presentation attack detection", in "The IEEE Winter Conference on Applications of Computer Vision", pp. 2635–2644 (2020).

Yang, B., Y. Ding, Y. Jin and K. Hao, "Self-organized swarm robot for target search and trapping inspired by bacterial chemotaxis", Robotics and Autonomous Systems **72**, 83–92 (2015).

Yang, Y., R. J. G. Clément, S. Ghirlanda and M. Porfiri, "A comparison of individual learning and social learning in zebrafish through an ethorobotics approach", Frontiers in Robotics and AI **6**, 71 (2019).

Zhu, L. and N. Laptev, "Deep and confident prediction for time series at Uber", in "2017 IEEE Intl. Conf. on Data Mining Workshops (ICDMW)", pp. 103–110 (2017).