Anticipation Strategies based on Artificial Intelligence Techniques for Communication between Two Groups of Agents

A Thesis

Presented in Partial Fulfillment of the Requirements for the

Degree of Master of Science

with a

Major in Computer Science

in the

College of Graduate Studies

University of Idaho

by

Juan F. Marulanda

December 2013

Major Professor: Terence Soule, Ph.D.

Authorization to Submit Thesis

This thesis of Juan Felipe Marulanda, submitted for the degree of Master of Science with a major in Computer Science and titled "Anticipation Strategies based on Artificial Intelligence Techniques for Communication between Two Groups of Agents," has been reviewed in final form. Permission, as indicated by the signatures and dates given below, is now granted to submit final copies to the College of Graduate Studies for approval.

Major Professor ____________________ Date __________
Terence Soule, Ph.D.

Committee Members ____________________ Date __________
Robert Heckendorn, Ph.D.

____________________ Date __________
Dean Edwards, Ph.D.

Department Administrator ____________________ Date __________
Gregory Donohoe, Ph.D.

Discipline's College Dean ____________________ Date __________
Larry Stauffer, Ph.D.

Final Approval and Acceptance by the College of Graduate Studies

____________________ Date __________
Jie Chen, Ph.D.

Abstract

In this thesis, a set of anticipation strategies based on Artificial Intelligence techniques is presented. The goal of these strategies is to support the ability of a group of agents ability to synchronize their movements during a mission when the communication is inconsistent. The agents are divided into two main groups: the leader and the followers. In general, the followers receive the leader's messages which informs them about its current status. The followers change their behaviors based on these incoming messages in order to complete the mission successfully. Scenarios are added where the message frequency is reduced. The followers are able to detect when messages are missing and use an anticipation module to try to infer the missing information. The anticipation module does not use the complete history of messages sent by the leader, instead it uses only a small window of recent messages. In addition, two environments were developed to test anticipation for this thesis: a simulated environment and a real environment using mobile robots. For the simulation, five different anticipation models (4 Neural Networks and one Fuzzy Logic) were tested using eight different leader behaviors. For the real environment, one anticipation model (Fuzzy Logic) was tested using three different leader behaviors. The results show that these anticipation strategies are generally capable of generating anticipated messages to fill in when leader messages are missing, thereby allowing the followers to complete missions successfully. In the simulation, three out of five anticipation models (2 Neural Networks and the fuzzy Logic models) were able to successfully reduce the meeting point error in all TS behaviors cases and under the most extreme message frequency conditions with an average error correction above 93%.

Acknowledgements

I would like to thank the Office of Naval Research for their support and funding through "Magnetic Signature Assessment System (MSA) using Multiple Autonomous Underwater Vehicles (AUVs) Phase 3", ONR N00014-10-1-0883.

Also, I would like to thank John Canning and Bryce Gill for their advisement on the communication system that the AUVs used in the MSA and Mine-Countermeasure (MCM) missions which helped to develop the simulated environment for this behavior, and Joshua Rubini for his suggestions related to Artificial Intelligence which were used for the anticipation module.

Finally, I would like to thank Michael Stinger and Travis DeVault for their contributions on the image processing and object detection modules that were used in the robots.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

Robotic agents are widely used in different fields and for many different tasks because their ability to perform these tasks autonomously. However, their behavior is generally limited to a set of fixed tasks. Autonomous agents are severely hampered by their inability to effectively and efficiently adapt to novel or changing environments, conditions, and missions. Agents are capable of operating autonomously with considerable success, which demonstrates the potential of autonomous agents, but the successes are within limited and accurately pre-defined environments and missions. More widespread, effective use of autonomous agents will remain out of reach until they are capable of autonomously adapting to changing conditions more likely as an agent capable of thinking ahead during particular scenarios. Technically, an adaptive autonomous agent can support a dynamic environment which means that it can fit better in a real open world situation. For this reason, anticipation was selected as the focus of research for this thesis.

In general, an agent that has a good knowledge of its environment and mission is capable of behaving "intelligently" i.e., responding flexibly to changing conditions and choosing behaviors in order to be one step ahead of possible adversities. An agent that is capable of generating, utilizing, and when necessary adapting or replacing the model of its behavior with a more accurate one, would represent a significant advance in autonomous agents. However, enabling this kind of dynamic models introduces a number of difficulties and risks; first, how to identify when the current model is failing; second, adapting the current model effectively and efficiently; third, generating and implementing novel models safely. These three features are also the basic principles when humans try to anticipate a particular situation by creating assumptions and using them at the appropriate time.

Robotics is a great example where these difficulties and risks happen often. Here, the robot has to capture and analyze the available information from the outer world in order to interact with it and to make decisions according to what the robot understands about the environment. However, a robot can sometimes find it difficult to interact with the environment because of its own hardware. A robot with bad sensors has poor performance because it captures noisy information from the environment or loses part of the information when the data is digitized. Then, it is necessary to use a model that can identify when the input information is outside the regular parameters for the robot and try to generate the correct information to allow the robot to continue with its normal routine. This can be compared to a behavior in humans where we analyze a particular situation and infer, based on previous information, future behavior even when there is missing information. For example, when a human is watching a car driving on a road but there is an object ahead that will block the car from the human's view for a period of time. Based on previous data, like the current speed of the car and the dimension of the blocking object, the human brain can easily infer when the car will appear on the other side of the object. Once the car is behind the object, the human is not able to see it, but he still can make accurate assumptions about the car's status. In this case, the human uses the sense of sight to obtain information about the car's behavior. Once the car is out of his sight, it can be considered that the communication link is interrupted and anticipation must be used when this interruption is detected to support the continuity of receiver's behavior (in this case the human). In [7], Greene et al. show a clear view of the resourcefulness of the brain. According to the authors, the part of the brain in charge of anticipation/inference is the hippocampus. There are two important functions that the hippocampus uses to perform anticipation: it encodes and retrieves episodic memories; and it is able to relate learned tasks to current events. This way, by using previous knowledge of a particular task, the brain is able to infer missing information in the context of the new task, like filling in missing words in a conversation. From a psychological perspective, the authors in [6] studied how the humans make inferences from reading.

They emphasize that the test subjects, after comprehending and gathering enough information from their reading, were able to create a reference model from the text and make diverse and rich inferences about it. Nevertheless, the authors state that these inferences can get very close to the text narration, but they will never be an exact replica of it. Another psychological approach was made in [19], where the authors studied the accuracy of the human inferences while analyzing meaningful and abstract environments. For example, as a meaningful environment, the subjects were told to predict a loan applicant's "credit score" based on "average monthly debt" and "average number of creditors". The abstract environment contained very poor information. Based on these studies, anticipation always requires access to previous knowledge of an environment or tasks in order to generate a correct or reasonably accurate answer. The accuracy of the answer might vary but this inferred answer will be close to the correct answer. Thus, sometimes it is not necessary to communicate every word that describes a task or environment to understand the main idea.

The behavior of the machines has been a focus of research the past decades. By taking ideas from the real world, machines are designed to mimic some kinds of human or animal behaviors in order to generate "intelligent" reactions. Artificial Intelligence attempts to exploit some of these behaviors. One behavior in particular is anticipation. This is a behavior that humans and some animals use as a reaction to their environment in order to interact with it, not only reflexively, but also as an action made before the information has stopped being received from the environment. This project will focus on the anticipation behavior in particular. Is the target changing its behavior while it is not visible? What information is necessary to generate an accurate answer? These are questions that the human brain analyzes by instinct. But machines are not as evolved as the human brain, so it is important to fully understand how anticipation works in order to recreate it in machines.

In these experiments, the main goal for the robots is to meet at a marked target point at the same time. One robot is able to choose its own speed but the other robots have to control their speeds based on the first robot's behavior. To accomplish this, they establish communications between them in order to maintain similar navigation by matching their speeds. This communication is a one way link, which means that the listener depends completely on the leader in order to complete the mission successfully. Like in any real environment, noise can be present and the communication can be intermittent. The listener must have the capability to detect this situation and infer what the leader is probably doing when a message is not received. Then, the listener must take action and change its behavior based on this inferred message.

This thesis is divided as follows: Chapter 2 presents an overview of the state of art related to this research as part of the work done at the University of Idaho and related to anticipation as a field of work. Chapter 3 introduces an article that was presented at the OCEANS'13 MTS/IEEE conference on anticipation strategies for lineal behaviors. Chapter 4 presents closer look at anticipation strategies on wider and more complex behaviors. This chapter includes additional and more complex experiments used with the same model described in chapter 3. Chapter 5 describes in detail the hardware experiments that were generated to test the anticipation strategies on Commodity Off the Shelf (COTS) robots. By combining Chapter 3, 4 and Chapter 5, an additional paper will be submitted to a scientific journal. Finally, Chapter 6 discusses general conclusions about the research presented in this thesis.

Chapter 2: Background

This project is part of a larger project funded by Office of Naval Research. It focuses on Autonomous Underwater Vehicles (AUV) for multiple tasks including mine-countermeasure (MCM) and magnetic signatures assessment (MSA) missions [10, 8, 12, 9, 21, 20, 13]. These projects have been developed at University of Idaho (UI) by an interdisciplinary group of research laboratories that includes the Laboratory for Artificial Intelligence and Robotics (LAIR). MCM and MSA missions use a set of tow to five AUVs that require constant and accurate communication in order to maintain group coordination. A major concern is the low bandwidth and potentially unreliable nature of underwater communications. Standard underwater acoustic communications work on a low frequency bandwidth with an operational range between 10Hz and 1kHz [24, 18]. Outside this range, underwater communication is not frequently used because it is much less reliable.

To address the problems of low bandwidth and unreliable communication, a subsystem that creates anticipated messages was inserted into the main system. As a strategy, anticipation has been used before for other types of problems such as imitating the natural language of humans to correct acoustic messages between Unmanned Underwater Vehicles (UUV) [11] and helping a simulated robot used inside a video game anticipate player movement [17]. Both approaches use previous, but not necessarily full, knowledge about the desired behavior of the robot to increase the anticipation performance during the tests. Although these two examples are describing anticipation strategies, the anticipation system that was implemented in this project does not use the same approach. The anticipation used in [11] is based on a linguistic logic that analyses the structure of a binary string using syntactic, semantic, pragmatical and behavioral logic. The anticipation used in [17] is a strategy based on a set of general rules that allows the robot to generate a plan to effectively ambush the other player in specific situations. Both strategies are based on static rules and use several variables to get the information from the environment to make a decision.

In other approaches to anticipation ([2], [25]), different strategies were used as a predictor for stock price behavior. Each article describes a different method to solve the stock prediction problem. In [2], the authors solve this problem by using a neural network and in [25], the authors use fuzzy logic model. Both articles used real stock prices values as input data in order to predict future behaviors. The results showed that these models are capable of solving the stock prediction problem. Other authors used similar approaches to solve this problem, including a neuro-fuzzy model in [1] and a model based on support vector machines (SVM) in [15]. The research supports the idea that anticipation based on AI techniques can be an effective and promising method, although it has not been fully exploited using robotic agents.

Robotics has recently become the focus of research where different anticipation strategies are used as an approach to solve the problem of making a robot accurately anticipate an event. For example, an anticipation method based on image processing was implemented by Koppula and Saxena [16] in order to detect the next possible move that a human was likely to do. Based on a couple of images, the robot was able to detect a particular action of the subject by recognizing his body and other objects in the environment and then, by analyzing the sequence, it was able to predict the next action of the subject. Another approach, where anticipation is used, was implemented by Satake et al. ([23]) to aid pedestrians. The robot tries to anticipate the future position of the pedestrian and his behavior and then, it tries to meet him at a estimated point on their path. This anticipation strategy uses a camera and a set of motion sensor to detect a person in the environment. Although the author tries to focus more on the human-robot social interaction, he describes the anticipation model properly. Just like in [16], the author in [23] uses image processing as a part of the anticipation strategy and both require a predefined model to generate an anticipated behavior. It is clear that any living creature that manifests an anticipation behavior, follows the same principle where it would need to know about the behavior of its target in order to decide what to do. These two articles start from the same principle and it is the same principle that is used in this project.

Another field that has started to become more common in robotics is live interaction between humans and robots. Humans have always tried to find better and easier ways to communicate with robots which includes the ability of the robot to make itself understood by humans. This means that both sides have to share a common language that they can comprehend and use properly. A promising approach is described in [4] where the authors show that a robot can learn and establish correct communications with a human. The robot learns to follow some syntax rules and identify what the human is trying to say. It uses this information to make a decision and take an action. We follow a more basic linguistic method to wrap the communication between agents followed by the control strategy that depends on this communication.

Two anticipation strategies were used in this project in the simulation phase. One uses an Artificial Neural Network module and the other uses a Fuzzy Logic module. Both approaches include flexible rules, limited information from the environment, and one of the models has learning capabilities. These two models are explained in detail in chapters 3 and 4. Using a similar configuration for the anticipation module, an additional model was implemented in a real-world environment using a set of robotic vehicles driving on the ground. Chapter 5 describes the implementation of this model in detail. In this second phase of the project, it was possible to test how the anticipation strategy was able to overcome a nosier environment (in the real world) to make the two robots complete the designated task. In this case, the physical sensors introduce new and unexpected errors in the system. Normally, the robot controller is able to regulate this type of problem but the communication between the robots must be functional at all times. When one of the system inputs fails, the anticipation module must pair with it to keep the robots synchronized by filling the gap that the failed system input left. Introducing more noise in the testing environment, like the noise created by the physical sensors, would create a more complex problem where the robustness of the anticipation module can be further tested.

Chapter 3: Learned Anticipation Strategy for Speed Control in an AUV Fleet

This paper was presented at the OCEANS'13 MTS/IEEE Conference in San Diego, CA on September 23rd 2013. This is an international conference which is sponsored by the IEEE Oceanic Engineering Society (OES) and the Marine Technology Society (MTS). The content of this chapter was slightly modified for this thesis to fit its format and to match with the rest of its content.

3.1 Abstract

Researchers at the Laboratory of Artificial Intelligence and Robotics (LAIR) and the Center for Intelligent Systems Research (CISR) from the University of Idaho (UI) have developed a message anticipation module for use by members of a fleet of autonomous underwater vehicles (AUV). The test scenario is a magnetic signature assessment (MSA) task, in which a fleet of five AUVs must simultaneously pass under a moving Target Ship (TS) at a predetermined location. During the task the TS informs the AUVs regarding its progress, allowing the AUVs to meet the TS at that measurement point despite variations in the TS's velocity. However, the underwater acoustic modems used by the actual AUVs are both low bandwidth and noisy. Thus, messages from the TS may be infrequent or erroneous. The goal of the anticipation module is to anticipate the TS's messages and, when necessary, use the anticipated message to fill in gaps left by dropped or erroneous messages. Successful anticipation depends on an agent having a good knowledge of its environment and mission. Research has shown that anticipation of words and sentences is central to human communication and language understanding. An agent that utilizes similar anticipation methods and is capable of using artificial intelligence techniques to generate, utilize, and adapt its message anticipation module with a more accurate one, would represent a significant

advance in autonomous agents. Five different anticipation models were created, four of them are based on a neural network model and one of them is based on a fuzzy logic controller model. In order to test effectiveness, robustness, and adaptability of the models used in the anticipation module, multiple tests were conducted in which the behavior of the target ship and the gap between messages were varied. All of the tested anticipation models were able to significantly reduce the error in the meeting point when there was a gap between messages. However, anticipation models without feedback produced better results than models with feedback; which suggests that using feedback may magnify the errors while trying to predict future messages.

3.2 Introduction

Currently, autonomous agents are severely hampered by their inability to effectively and efficiently address novel, changing, or noisy environments, conditions, and missions. Agents are capable of operating autonomously with considerable success, which demonstrates the potential of autonomous agents, but the successes are within limited and accurately pre-defined, and relatively noise-free, environments and missions. More widespread, effective use of autonomous agents will remain out of reach until they are capable of autonomously adapting to noisy and changing conditions.

A common difficulty for autonomous vehicles acting in concert is lost or erroneous messages, either due to communication errors or to human error. Such communication difficulties lead to unexpected changes in the environment (e.g. if the agent receives incorrect information about the environment or fails to receive important information) and create a noisy operating environment. Thus, the goal of this research is to develop a method by which agents can learn to address the particular problem of noisy and/or low bandwidth communications. Our approach is to include an *anticipation* module that allows autonomous agents to anticipate future messages based on previous messages and current mission conditions. When messages are lost or erroneous, the anticipated messages can be used to either fill in for the missing message or help correct erroneous messages.

This chapter is divided as follows: Section 3.3 presents a brief overview of previous work and research which the ideas in this paper are based on. Section 3.4 presents a description of the simulated environment used to recreate a magnetic signature mission using 5 AUVs. Section 3.5 presents two different artificial intelligence methods (artificial neural networks and fuzzy logic) that solve the communication problem recreated between the group of submarines and the ship. Section 3.6 presents the 3 behaviors and 3 communication situations used to test the accuracy and robustness of the system. Finally, section 3.7 presents some relevant conclusions of the project and the results.

3.3 Background

This project is part of an on-going project on using Autonomous Underwater Vehicles for multiple tasks including mine-countermeasure (MCM) and magnetic signatures assessment (MSA) missions which are being developed at University of Idaho (UI) ([10, 8, 12, 9, 21, 20, 13]). Because these tasks require coordination between multiple (2-5) AUVs, accurate communications are critical. Thus, a major concern is the low bandwidth and potentially unreliable nature of underwater communications. In this paper, we test the use of anticipated messages as a mean to overcome the limitations of the communication system. Anticipation is a strategy that has been tried before for other types of problems including trying to imitate the natural language of humans in order to correct acoustic messages between Unmanned Underwater Vehicles (UUV) [11] and helping a simulated robot used inside a video game anticipate player movement [17]. Both approaches use previous, but not necessarily full, knowledge about the desired behavior of the model to increase prediction accuracy during the tests. However, the anticipation strategies are not the same as are used in this paper. The anticipation used in [11] is based on a linguistic logic which analyses the structure of a binary string using syntactic, semantic, pragmatical and behavioral logic. In [17], the anticipation strategy is based on a set of general rules which allows the robot to generate a plan in order to ambush the other player in specific situations effectively. Both strategies are based on static rules and use several variables to get the information from the environment to make a decision. In this research, we present two additional strategies for anticipation, which use flexible rules, limited information from the environment, and one of the models is capable of learning.

In [2] and [25], anticipation is used to predict a stock price behavior. Each article describes a different strategy to solve this problem: [2] solves this problem by using a neural network and [25] uses fuzzy logic model. Both articles used real stock prices values as input data in order to predict future behaviors. The results showed that these models are capable of solving this type of anticipation problem. Other similar approaches to solve this problem also use a neuro-fuzzy model [1] and a model based on support vector machines (SVM) [15]. This work supports the idea that anticipation is a promising approach although in that research it was not applied to robotic agents.

Following these ideas the Center for Intelligent Systems Research (CISR) at UI has developed a computational architecture called Language-Centered Intelligence (LCI) which allows autonomous agents to reason hypothetically about their environment and mission via "anticipated" observations [10]. These anticipated observations guide the agent in its mission and serve, when compared to actual, future observations, to measure the accuracy of the current model, thus, mitigating the risk of identifying a failure in current model.

The test problem in this research is a Magnetic Signature Assessment (MSA) mission which uses a fleet of AUVs to record the magnetic signature of a target ship (TS). The AUVs must pass under the TS in a pre-defined measurement zone, allowing them to measure the TS's magnetic signature. In a full mission the TS makes two to four passes through a measurement zone; if two passes, then one East/West pass and one North/South, and if four, then one pass in each of the cardinal directions. During each pass through the measurement zone, the fleet of AUVs passes underneath the ship. The system assumes that there is a sufficient number of AUVs to capture all of the necessary data for each direction with a single pass and that the AUV fleet approaches the ship from the opposite heading (i.e., bow to bow). The simulation used in this research to test the anticipation models is based on one pass in the MSM.

3.4 Environment

The simulation environment was developed using C#[1] and the AForge[2] library for the AI resources. The environment simulates one pass of the MSA mission with a focus on the control of the group of AUVs using the communication link with the TS. In the simulation, the TS regularly broadcasts its current progress toward the measurement box to the AUVs. Using this information, the AUVs are able to adjust their current speed in order to maintain formation and reach the meeting point at the same time as the TS. The metric of success is how close the AUVs are to the center of the measurement box when they pass under the TS. A fuzzy logic controller on each AUV is in charge of this function, it uses the current progress of the AUV and the message that it receives from the TA to determine speed. The heading of the AUVs is determined by 3 sets of waypoints, which are used to calculate the current progress of the AUVs (the sets of ovals in Figure 3.1). To increase the complexity and realism of the problem, the AUVs are assigned random starting positions and headings in each run. Figure 3.1 shows a group of AUVs and the ship at the start of a mission run. Figure 3.1 also shows the waypoints along the path of the AUVs (ovals).

The TS can be configured with any of 3 different behaviors that, in general, make

[1]Microsoft Visual C# 2010 Express
[2]AForge .Net Framework 2.2.4

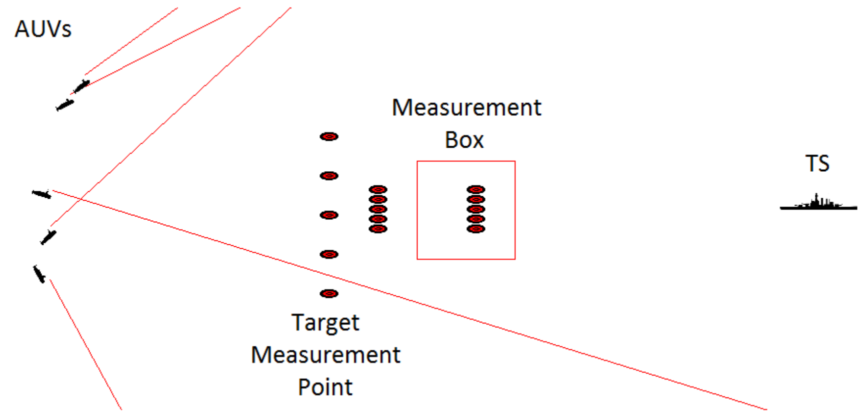


Figure 3.1: Simulation environment for the anticipation tests showing a typical starting configuration. The five AUVs are on the left, their initial positions are semi-randomized and their headings are random. The target ship (TS) is on the right, it always begins with the same position and heading. Ovals are the AUVs' waypoints.

the TS change its speed and/or its acceleration in different sections of a mission. This behavior forces the AUVs to vary their speed accordingly, in order to reach the measurement point at the same time as the TS. The TS behaviors are described in detail later in Section 3.6. The AUVs initially have no information about which behavior was chosen for TS, thus they must rely on the messages from the TS to reach the measurement point at the same time as the TS.

3.5 Methods

The simulation determines how the TS behaves and how often the AUVs receive messages from the TS regarding its progress. The TS uses a fuzzy logic controller to control its speed. The controller calculates fuzzy membership in five variables that measure progress. These five fuzzy values are sent as the message from the TS to the AUVs. The five fuzzy variables are: way behind, behind, on schedule, ahead, or way ahead, and are calculated based on the initial, scheduled meeting time (Figure 3.2). Each variable can have a value between 0 and 1, but, because the message is informing about a specific state of the TS, there can be only at most 2 variables whose value is greater than 0.

Way behind	Behind	On Schedule	Ahead	Way Ahead
[0...1]	[0...1]	[0...1]	[0...1]	[0...1]

Figure 3.2: Messages from the target ship (TS) consist of five fuzzy values.

The ship sends messages at specific, adjustable intervals. Longer intervals represent limited bandwidth or a noisy environment where messages often do not arrive. In a situation like this, the AUVs do not receive updated values for the TS's position at each time step. Their default solution is to assume that the ship is returning to the initial schedule, and the AUVs' fuzzy controllers attempt to return to the schedule as well.

Our alternative approach is based on anticipation. An anticipation module in each AUV attempts to generate an anticipated message when a message from the TS is missing. The anticipation module is configured to create a new anticipated message from the TS based on a set of the previous messages received from the ship. When a message is missing, the anticipated message is used by the AUV as the regular message allowing the AUV to update its current velocity by anticipating the TS next message.

3.5.1 Anticipation

Anticipation is performed by using a list of prior messages as inputs to predict a missing message. This list includes a fixed number of recent messages, thus the anticipation module works with recent information and not the overall record (Figure 3.3). Every time the list is updated with a new message from the ship, the oldest message is removed from it, thus the list of previous messages used to anticipate future messages is a FIFO list.

Several models for the anticipation module based on artificial neural networks and fuzzy logic were tested to determine which is able to anticipate future messages most

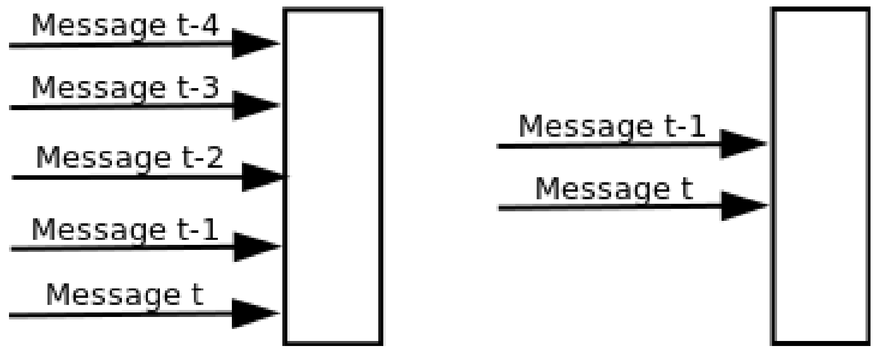


Figure 3.3: Message-list structure: the anticipation module uses a history of previous to generate the possible next message. The Neural Network models were trained using both structures; the fuzzy logic model use only the structure with 2-messages input

accurately. Either 2 or 5 previous messages are used, to test whether a model can produce better results if it uses a longer message history to anticipate future messages.

3.5.2 Neural Networks

To test the robustness of the neural network approach and learning algorithms 4 models were compared. The number of messages used as inputs and the numbers of neurons in the single hidden layer were modified to measure the resulting behavioral changes (Figure 3.4). The following combinations were tested:

- 2 messages (10 inputs) and 5 neurons in the hidden layer.
- 2 messages (10 inputs) and 10 neurons in the hidden layer.
- 5 messages (25 inputs) and 5 neurons in the hidden layer.
- 5 messages (25 inputs) and 10 neurons in the hidden layer.

The neural networks were trained using back-propagation ([14] and [22]). 10000 epochs were used in the training phase of each neural network. Training data consisted of messages from the 3 TS scenarios described below. Note that a single neural

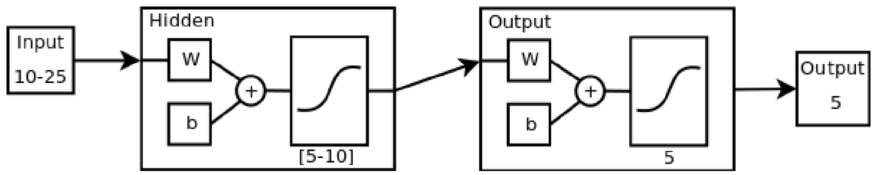
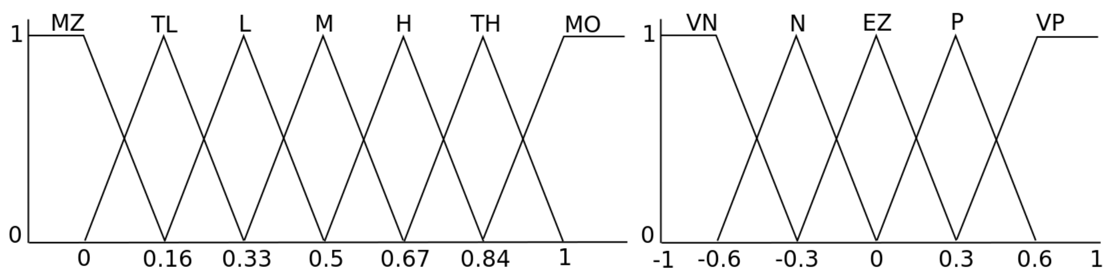


Figure 3.4: General Neural Network structure. The network has 10 or 25 inputs, depending on the size of the message history, a single hidden layer, and 5 outputs, one for each value in the anticipated message. A sigmoid function is used for the activation function.



(a) Variable Input (b) Fuzzy Difference Input

Figure 3.5: Fuzzy Sets that describes the current current value of one variable inside the message and the difference that it has with it previous value

network was trained on and tested on all 3 cases, so it had to learn to generalized across the 3 test behaviors of the TS.

3.5.3 Fuzzy Logic

One fuzzy logic model was tested. 2 previous messages are used as the inputs for this model [14]. The two previous values of each of the 5 fuzzy values is used to calculate the next, anticipated, value. E.g. the two previous values of the first fuzzy variable are used to anticipate the future value, by evaluating the last received value and the difference between its the last received value and the value before that. Using the two previous values allows the anticipation module to determine how fast each fuzzy value is increasing or decreasing as part of the anticipation process.

The value of each variable in a message is assigned a membership in 7 fuzzy sets: Zero (MZ), Too Low (TL), Low (L), Medium (M), High (H), Too High (TH) and

Table 3.1: General fuzzy rules

Error \ Message	*MZ*	*TL*	*L*	*M*	*H*	*TH*	*MO*
VN	MZ	MZ	TL	TL	L	M	MO
N	MZ	TL	L	L	M	H	MO
EZ	MZ	TL	L	M	H	TH	MO
P	MZ	L	M	H	H	MO	MO
VP	MZ	M	H	TH	TH	MO	MO

One (MO) (Figure 3.5a). The change in value of each variable (the difference between the previous two values in a message) is assigned a membership in 5 fuzzy sets: Very Negative (VN), Negative (N), Zero (EZ), Positive (P), Very Positive (VP) (Figure 3.5b). Based on these sets, a group of basic rules were created, defined in as a fuzzy associative matrix: Table 3.1.

Basically, the fuzzy logic module takes each of the 5 values inside the most recent message (Figure 3.2) and compare it with the value from message before that by applying the rules in Table 3.1. These rules are used to anticipate the next value for each of the five variables in a TS message.

An additional small set of rules was added to the fuzzy logic module to allow it to create a crossed relationship between the values in a message. The additional fuzzy rules associate the anticipated value of a message value m_i with its neighboring message values. These fuzzy rules are used to check if a value in a message has a current value of 0 and if one of its neighbors (message values m_{i-1} or m_{i+1}) are close to their medium value and decreasing. If so, it can be anticipated that the message value m_i will be about to change. The actual fuzzy rules are:

$\mathrm{TL}_{m_i} = \mathrm{MZ}_{m_i} \&\& M_{m_{i+1}} \&\& (N_{Ei+1} || \mathrm{VN}_{Ei+1})$

$\mathrm{TL}_{m_i} = \mathrm{MZ}_{m_i} \&\& M_{m_{i-1}} \&\& (N_{Ei-1} || \mathrm{VN}_{Ei-1})$

These rules anticipated that if a message value's is currently zero (MZ), it's anticipated membership in the TL (two low) set depends on the neighboring messages values.

Note that the TS message describes the condition of the TS at most using 1 or 2 of the 5 variables in the message; the other 4 or 3 variables remains at 0. For example, if the TS is getting behind schedule, the values for the sets behind schedule and way behind schedule get progressively larger (Figure 3.2) while the other values remain at zero.

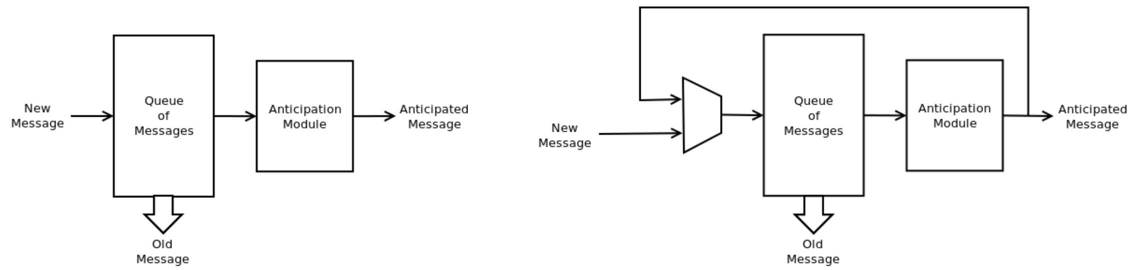
3.6 Results

To evaluate the anticipation module and its robustness, 3 different behaviors for the TS were created:

- *On schedule (OnS):* the TS maintains a constant velocity using its correct speed and reaches the measurement point on schedule.
- *0.8 speed (0.8S):* the TS maintains a constant velocity of 0.8 of its correct speed. Thus, the TS reaches the measurement points significantly behind schedule and the actual measurement, which take place when the AUVs and the TS pass each other, may occur significantly before (to the right of the measurement box in Figure 3.1) the measurement point.
- *1.2 speed (1.2S):* the TS maintains a constant velocity of 1.2 of its correct speed, and reaches the measurement point significantly ahead of schedule and the actual measurement, which take place when the AUVs and the TS pass each other, may occur significantly after (to the left of the measurement box in Figure 3.1) the measurement point.

The anticipation module was tested using 6 different strategies. No anticipation, 4 different neural networks, and a fuzzy logic model:

- *NA:* No Anticipation



(a) Without Feedback (b) With Feedback

Figure 3.6: Anticipation Strategies: the anticipation module can generate a brand new message every time a real message is inserted in the queue of messages or it can use a feedback to insert a new anticipated message in the queue every time a real message is missing which allows the module to generate a brand new message more often.

- *NN1:* 10 inputs (2 messages) and 5 neurons in the hidden layer.
- *NN2:* 10 inputs (2 messages) and 10 neurons in the hidden layer.
- *NN3:* 25 inputs (5 messages) and 5 neurons in the hidden layer.
- *NN4:* 25 inputs (5 messages) and 10 neurons in the hidden layer.
- *FL:* Fuzzy Logic, using 2 input messages.

The anticipation modules were tested using 2 different strategies:

- *Without feedback:* the module uses only the actual messages received from the TS (Figure 3.6a). When a message from the TS does not arrive, the anticipation module uses the last N (2 or 5) received messages to anticipate the next message. A list is used to store these messages and it is updated only when a new message from the TS is received.

- *With feedback:* the anticipation module treats anticipated messages as received messages (Figure 3.6b). The message list is updated every time a new message is received from the TS. But when a message does not arrive, the anticipated message is included in the message list that will be used the next time a message is anticipated.

Table 3.2: Absolute value of group meeting point error and standard deviation. Each anticipation model shows the results of running the simulation without feedback (top value), and with feedback (bottom value), respectively. For example, for the 0.8 behavior with 6 time steps per message (6S/M) and without anticipation, the TS and AUVs met 41.77 m away from the target point; with anticipation using no feedback and NN1, they met only 13.65 m away from the target point; and with anticipation using feedback and NN1, they met 14.95 m away from the target point.

Test \ **Model**		***NA***	***NN1***	***NN2***	***NN3***	***NN4***	***FL***
OnS	1 S/M	1.06(0.27)	1.06(0.28) 1.06(0.28)	1.06(0.28) 1.06(0.28)	1.06(0.27) 1.06(0.27)	1.06(0.28) 1.06(0.27)	1.06(0.28) 1.06(0.28)
	2 S/M	0.70(0.51)	0.89(0.26) 0.87(0.28)	1.01(0.28) 1.02(0.27)	1.02(0.27) 1.02(0.26)	1.00(0.27) 0.97(0.27)	1.16(0.28) 1.00(0.26)
	6 S/M	0.70(0.62)	0.75(0.28) 0.51(0.63)	0.99(0.27) 0.66(0.28)	1.00(0.27) 0.82(0.74)	0.93(0.28) 0.70(0.62)	1.21(0.29) 0.61(0.26)
0.8S	1 S/M	10.98(0.18)	10.98(0.18) 10.98(0.18)	10.98(0.18) 10.98(0.18)	10.98(0.18) 10.98(0.18)	10.98(0.18) 10.98(0.18)	10.98(0.18) 10.18(0.28)
	2 S/M	28.30(0.27)	12.66(0.32) 12.66(0.19)	12.56(0.46) 12.27(0.19)	11.92(0.44) 11.95(0.44)	12.68(0.19) 12.43(0.46)	11.30(0.43) 11.30(0.43)
	6 S/M	41.77(0.26)	13.65(0.29) 14.98(0.28)	10.99(0.18) 14.90(0.31)	12.47(0.36) 13.74(0.06)	13.19(0.26) 15.65(0.42)	11.59(0.31) 14.05(0.25)
1.2S	1 S/M	1.09(0.26)	1.09(0.26) 1.09(0.26)	1.09(0.27) 1.09(0.26)	1.09(0.26) 1.09(0.26)	1.09(0.26) 1.09(0.26)	1.09(0.26) 1.09(0.27)
	2 S/M	16.25(0.08)	2.87(0.27) 2.77(0.29)	3.83(0.28) 3.77(0.27)	3.53(0.27) 3.96(0.27)	3.81(0.27) 4.00(0.28)	1.63(0.25) 2.63(0.51)
	6 S/M	31.67(0.07)	4.46(0.26) 6.09(0.49)	5.07(0.27) 8.60(0.15)	5.02(0.39) 18.87(0.04)	4.93(0.26) 7.60(0.19)	2.47(0.32) 10.57(0.14)

For training, AUVs received messages every time step, the optimal condition for all the TS behaviors. For testing the number of time-steps that the TS waits to send a message was varied. The three test cases were, 1 message per time step, one message every 2 time steps, and 1 message every 6 time steps.

Performance is judged by measuring the distance between the target measurement point and the actual meeting point between the TS and the AUVs. Each test consists of evaluating one anticipation model for each value of each experimental variable (steps/message, TS behavior). 10 trials were performed for each test, which gives a total of 3600 trials for the entire experiment, 720 trials for each anticipation model. Table 3.2 shows the average values for the meeting points between the group of AUVs and the ship. Lower values are better. The highlighted values represent the best results between the 5 anticipation models for each test.

The results with no anticipation (the column labeled NA in Table 3.2) show that when the TS is on schedule (rows 1, 2, and 3) the TS and AUVs meet close to the designated meeting point - just over 1 simulated meter away - despite the randomized starting positions and angles of the AUVs. This result confirms that the fuzzy controllers responsible for keeping the TS and AUVs on schedule perform correctly.

The results with no anticipation, when the TS is off schedule (rows marked 0.8S and 1.2S) show that when messages are received frequently (rows labeled 1M/S) the AUVs do fairly well even without anticipation, although when the TS is slow (row 0.8S) the are off by roughly 10 meters. However, as messages become increasingly infrequent (2S/M and 6S/M) AUVs with no anticipation fail to meet the TS near the designated meeting point, with larger message gaps leading to worse results.

Table 3.2 shows that all anticipation models showed a significant improvement over no anticipation, when the TS was off schedule and there were gaps between the messages. I.e. anticipation does, partially, and in many cases completely, makes up for the large gap between messages. In the worst case, a slow TS (0.8S) and infrequent messages (6S/M), AUVs without anticipation met the TS, on average, almost 42 meters from the designated measurement point. In contrast, AUVs with any of the anticipation models, on average, met the TS within 15 meters of the designated point.

Overall NN1 and the fuzzy logic models had better results on 5 of the test cases each. These two models showed the smallest error between the correct meeting point and the one obtained in the experiment. Model NN4 also had good results on most of the tests, but did not have the best results in any of the 18 configurations used for the experiment.

In general anticipation models using feedback performed slightly worse than models not using feedback.

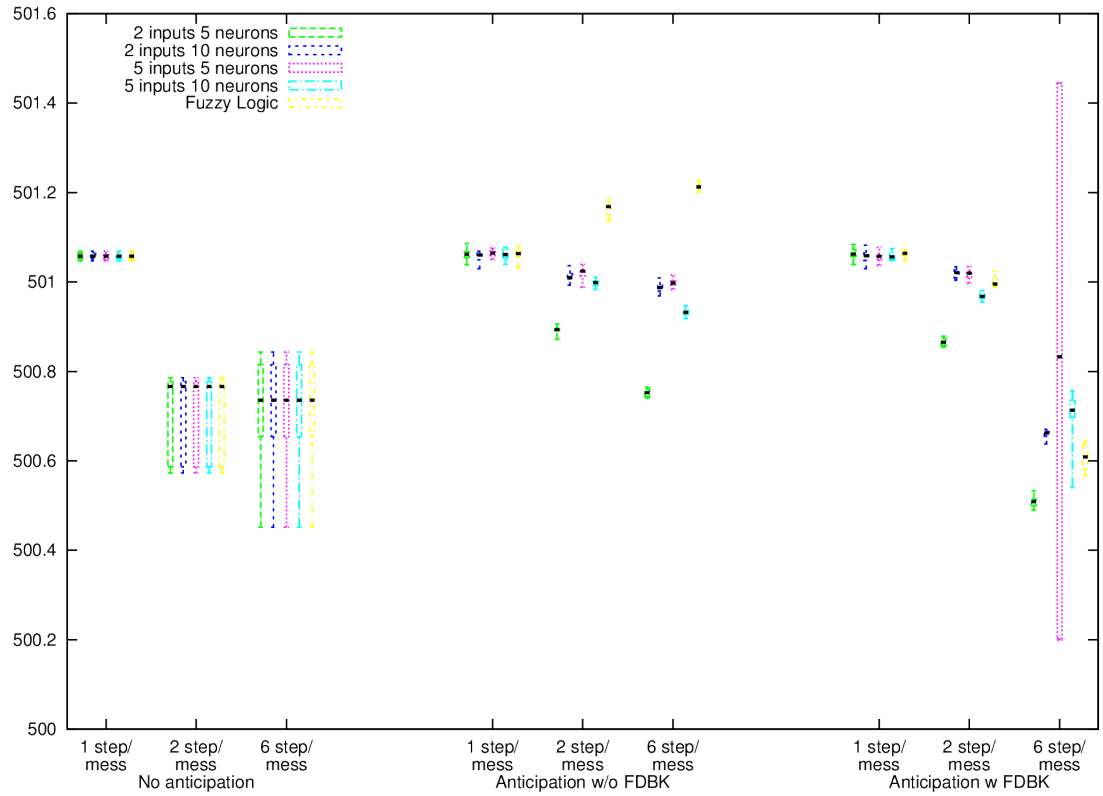


Figure 3.7: Average and quartile results with all anticipation models when the TS is on schedule. Overall results are similar, with small errors in the meeting point for all cases, but using anticipation reduces the variability in the meeting point, except for anticipation using the neural network with 5 input messages and 5 neurons in the hidden layer, with feedback, when there one message is received every 6 time steps.

Figures 3.7 and 3.8 show quartile plots for the results obtained from all the anticipation models for 2 TS behaviors: OnS and 0.8S. Each figure shows 3 main groups which represent the three types of simulations that were tested: No Anticipation, Anticipation with Feedback and Anticipation without Feedback. Within each group, there are three additional groups that represent the possible message gaps: 1 step/message, 2 steps/message, and 6 steps/message. Each of those groups has 5 elements which represent the anticipation models that were used for this experiment: NN1, NN2, NN3, NN4, and FL. Figure 3.7 shows that as the message gap increases there is generally wider variability in the meeting points. This is also true in Figure 3.8, but the change in scale obscures the dispersion. Figure 3.8 also shows how in-

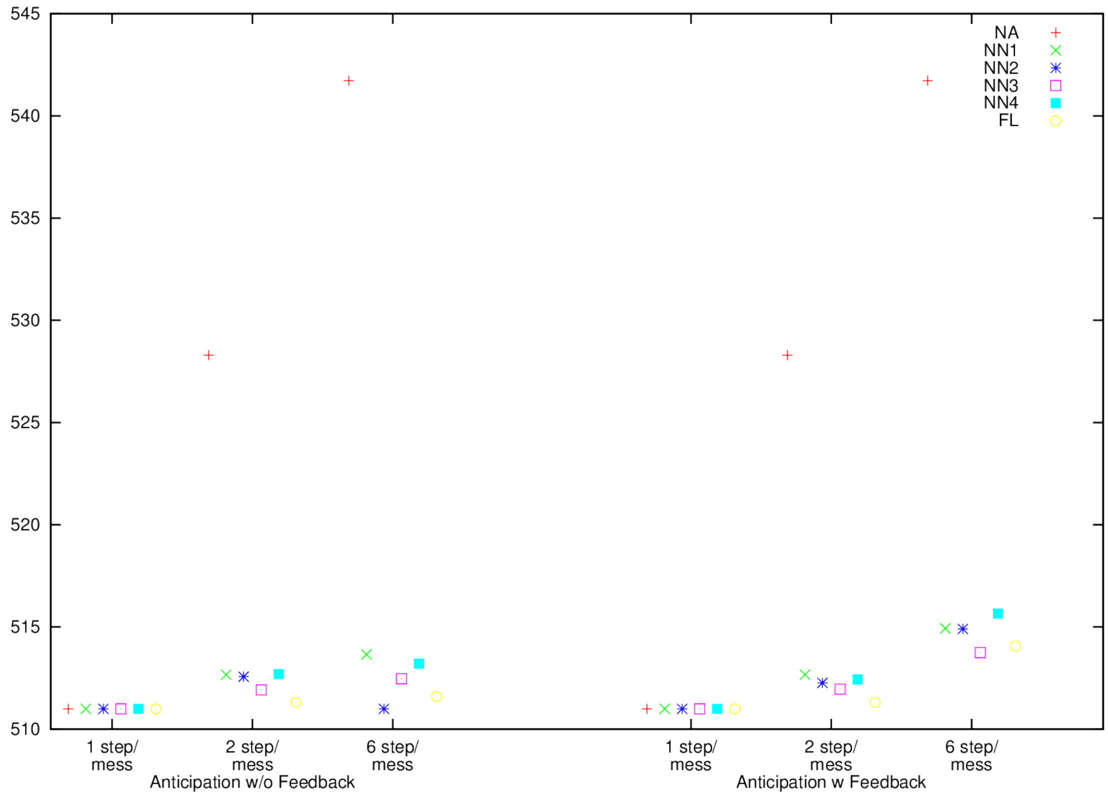


Figure 3.8: Global average of the meeting point results for the AUVs obtained with all anticipation models when the TS is traveling at 0.8 of its correct speed.

creasing the message gap significantly impacts the meeting point when there is no anticipation of the missing messages, but not with anticipation. Figure 3.7 suggests that the dispersion in the meeting points is generally reduced by using anticipation.

Figures 3.9, 3.10, 3.11 and 3.12 show the average position of the the individual AUVs (1-5) with NN1 and FL (which produced the best behaviors during the experiments). Figures 3.9 and 3.11 show the results when the TS behavior is OnS, and Figures 3.10 and 3.12 show the results for behavior 0.8S. These figures show that both anticipation models reduce the error for every AUV. Equally important these figures show that the errors for the individual AUVs are similar. Thus, in general the anticipation modules not only reduce the meeting point error, but also maintain the formation that the group has in optimal conditions.

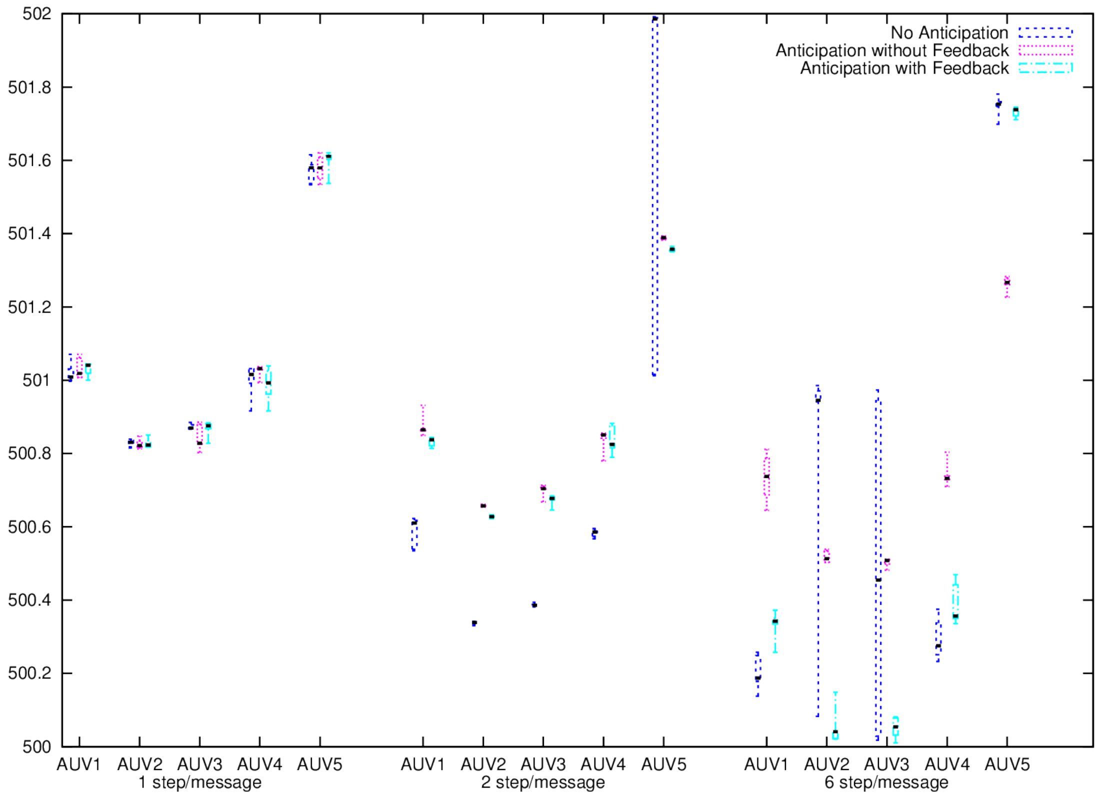


Figure 3.9: Average and quartile results for the meeting point by AUV, when using a neural network with 10 inputs and 5 neurons in the hidden layer when the TS is on schedule. AUV 5 tends to be slightly further from the measurement point. Variability in the AUV position, particularly for AUV 5, is highest with no anticipation.

Figure 3.9 shows that anticipation reduces the variation in the meeting point, keeping the AUVs in formation. For example, once the value of the steps/message increases, the variation in AUV position with no anticipation increases considerably, especially in AUV 5. The other 2 models, which use anticipation, sproduce much smaller variation in the AUVs' positions. For example, the variation for AUV 5 decreases considerably with both models compared to no anticipation. Additionally, anticipation helps the AUVs get to the meeting point in formation. Most notably the error with both anticipation models and 2 or 6 steps per message were similar to the results with no anticipation and one message per time step. This show that anticipation is successfully "filling in" the missing messages.

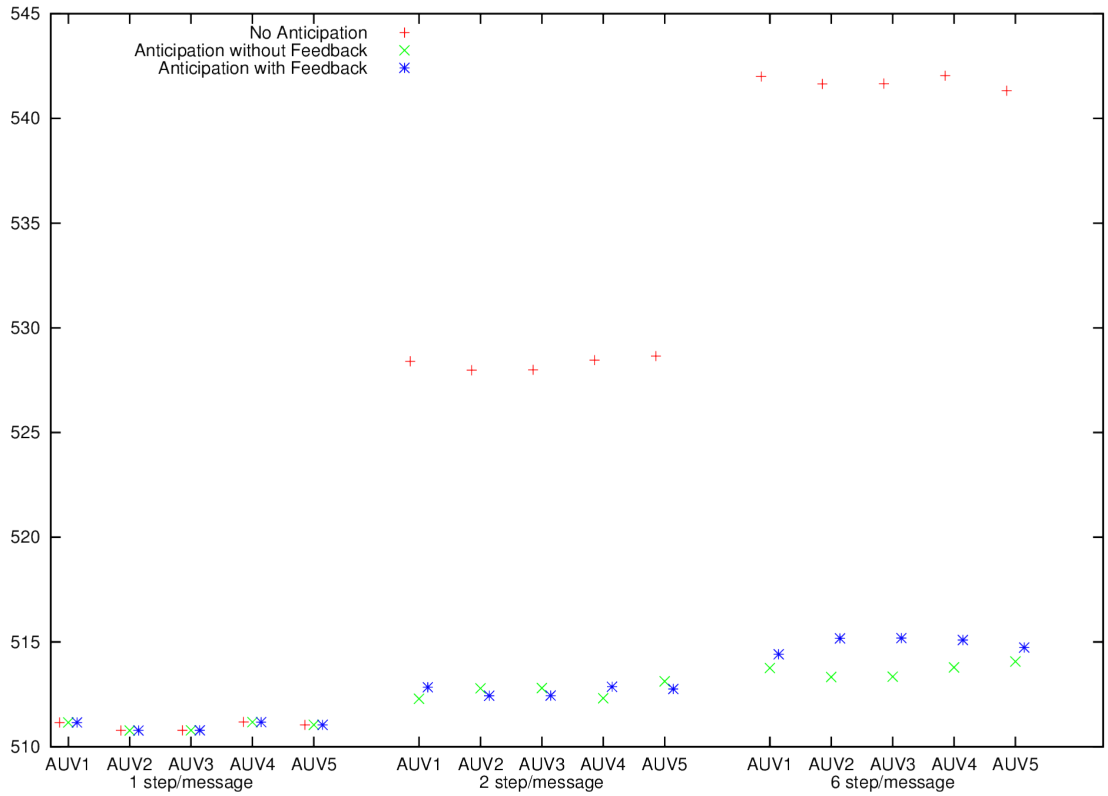


Figure 3.10: Average and quartile results for the meeting point by AUV, when using a neural network with 10 inputs and 5 neurons in the hidden layer when the TS is using 0.8 of its correct speed. All AUVs have similar errors, showing that they remain together. Errors are much larger when there is no anticipation and a messages are infrequent.

Figures 3.11 and 3.12 show the results for the same cases as in Figures 3.9 and 3.10 but using the fuzzy logic model. Figure 3.11 shows that the fuzzy logic model also reduces the variation in the AUVs' position when the anticipation is enabled, but slightly less than the neural network model. On the other hand, Figure 3.10 shows that the fuzzy logic model does a better job of maintaining the group formation, getting them closer, on average, to the correct meeting point.

3.7 Conclusions

The results shows that all of the anticipation models were able to significantly reduce the error in the meeting point when the TS was off schedule and messages were

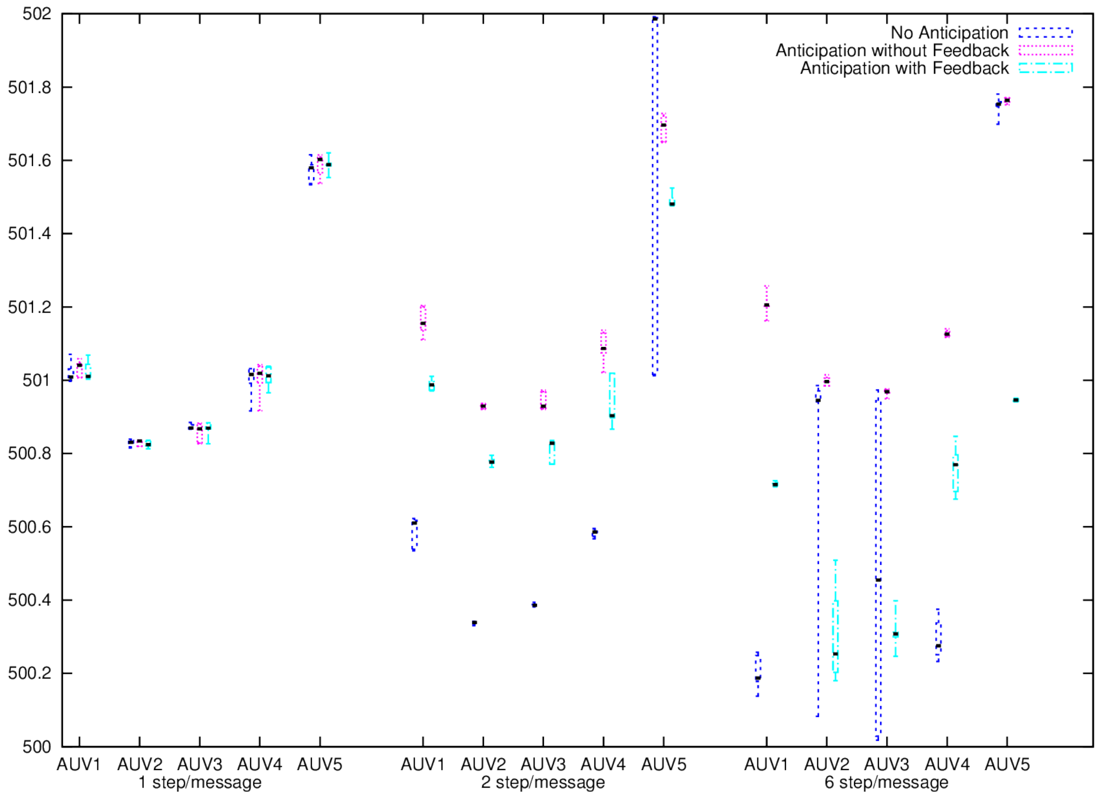


Figure 3.11: Individual average of the meeting point results obtained using fuzzy Logic when the TS is on schedule.

infrequent. This is a very promising result, as it strongly suggests that anticipation can be an effective method to address communication problems cause by noisy or low bandwidth communication channels. In our results NN1 model had, in general, the best performance followed by the fuzzy logic model. The other neural network NN2 and NN3 models had good results with values that were better than the NN1 and fuzzy logic models, but only under a few of the test cases. Both NN1 and the fuzzy logic model only used the two previous messages, suggesting that, at least for this problem, a short message history is sufficient to anticipate future messages.

The results (Figures 3.9, 3.10, 3.11 and 3.12) also show that anticipation models without feedback produced better results than models with feedback, although both models were successful at significantly reducing the error in the meeting point. This suggests that using feedback, i.e. using anticipated messages to predict future

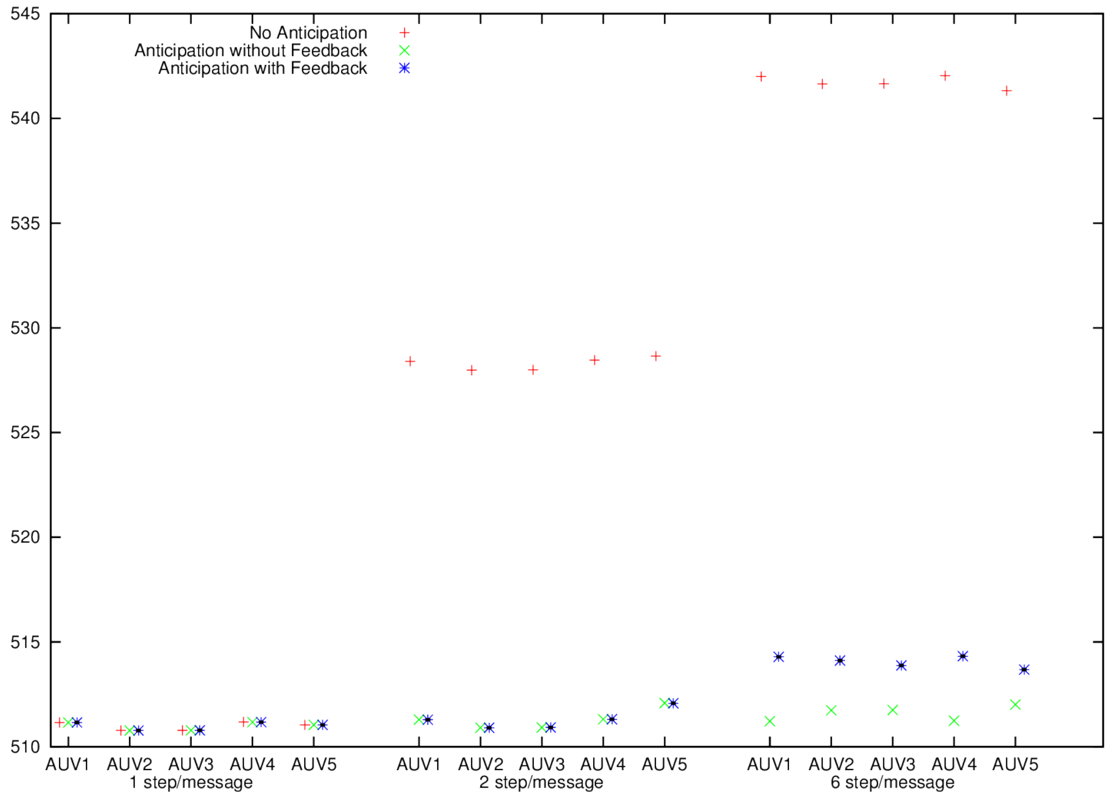


Figure 3.12: Individual average of the meeting point results obtained using fuzzy logic when the TS is using 0.8 of its correct speed.

messages, may magnify the errors in the anticipated messages.

All of the anticipation models also reduced the variation in the meeting position with the TS, but models with feedback had slightly larger variations than models without feedback. Overall, it is clear that the anticipation models presented here are effective at anticipating messages and that the anticipated messages can be successfully used in the place of lost messages (or messages that are forced to be discarded due to errors). Finally, it is worth noting that the test problem used in these experiments required a fleet of five autonomous underwater vehicles (AUVs) to reach a specific location, as a group, at a time determined by the behavior of another vehicle. This type of coordinated, group behavior with a dynamic goal represents a very general and useful behavior, thus the results of this research has potential benefits for a wide range of applications.

Chapter 4: Artificial Intelligence System as an Anticipation Strategy for the Speed Control of a Group of Submarines Based on Fuzzy Logic Communication

This Chapter contains the additional results from the anticipation problem for the AUV project. These results plus the ones from Chapter 5 will be submitted to a journal. The structure of this paper was modified to fit with the structure of the thesis and to avoid having duplicate sections with the previous chapter.

4.1 Introduction

This Chapter presents data from additional experiments performed with the simulated mission that were not included in the Oceans paper presented in Chapter 3. These experiments are basically the same as those in the previous chapter, but have more complex target ship behaviors. The goals of these experiments is to determine how robust the anticipation system can be by increasing the difficulty of the TS behavior. Also, it is important to determine which model works better over a whole set of problems. This means that the models are evaluated to see which one works better with various TS behaviors and TS message frequencies. This chapter does not present a section for the experimental environment and the methods as they have been described in Chapter 3.

This chapter is divided as follows: Section 4.2 presents a set of five additional TS behaviors with three different communication patterns which were used to test the accuracy and robustness of the entire system; it also describes the obtained results. Finally, section 4.3 presents some conclusions based on the results the results.

4.2 Results

The simulation, including the AUV behaviors, neural networks, fuzzy logic, the message format, the message frequencies are the same as in Chapter 3. Five new target ship behaviors are part of the simulation options. Three of these new behaviors were also used to train the Neural Networks and the remaining two behaviors were used for testing only. These additional behaviors were also created based on Figure 3.1.

The five TS behaviors are:

- *Slowdown with recovery of its schedule (SwR):* the TS starts moving at its regular speed until it get to position 800 and decreases its speed to 0.5 of its regular speed. Then, at position 700, the TS accelerates until it reaches a velocity of 1.5 of its regular speed. Finally, at position 600, the TS returns to its default velocity. During the stage between positions 700 and 600, the TS recovers the exact time and distance lost on the previous stage by using the opposite value for its speed.

- *Slowdown without recovery of its schedule (SwoR):* this behavior is similar to the previous one. The only difference is that the TS does not recover its schedule between position 700 and 600. First, the TS starts moving at its regular speed until it get to position 800 and decreases its speed to 0.5 of its regular speed. Then, at position 700, the TS accelerates until it reaches a velocity of 1.1 of its regular speed. Finally, at position 600, the TS returns to its default velocity.

- *Experimental (Exp)* the TS starts with its default speed until it reaches position 800. Here, it slows down to 0.8 of its regular speed and it maintains this speed until position 650. Then, it accelerates until it doubles its default speed. For this test, the acceleration is considered almost double the normal acceleration to get to the maximum speed faster than usual.

- *Test 1 (T1):* for this behavior, the TS has 5 stages where the velocity changes. First, the TS starts with its default speed until it reaches position 900. Here, the TS slows down until it reaches 0.5 of its regular speed. Once the TS passes position 800, it start to accelerate to reach 1.9 of its regular speed. After position 700, the TS slows down until it reaches 0.7 of its regular speed and finally, it increases its velocity to 1.4 of its regular speed after it passes position 600. It maintains a velocity of 1.4 its regular speed until it reaches the end.

- *Test 2 (T2):* this test also contains 5 stages as in the previous test. The difference is that, not only the values are changed but also the acceleration is suppressed from this test. Here, the speed has sudden changes. As in Test 1, the TS starts with is default speed until it reaches position 900, then its velocity is changed to 1.8 of its regular speed. At positions 800 and 700, the velocity of the TS is set to 0.9 and 1.3 of its regular speed respectively. In the last stage, the TS changes its velocity to 0.6 of its regular speed.

As in Chapter 3, the goal is to test the anticipation models by recording the meeting point between the TS and AUVs. Each test consists of evaluating one anticipation model by modifying the experimental variables (steps/message, TS behavior) with all the possible configurations. 10 trials were made for each test which gives a total of 3600 trials for the entire experiment. This means that, because five anticipation models are being tested, 720 trials are made for each anticipation model. Table 4.1 shows the average values for the meeting points between the group of AUVs and the ship. For each TS behavior, the data under optimal conditions was also included. It is consider as an optimal condition when no anticipation is being used and the message frequency is 1 S/M. To determine that the anticipation model has done a good job while the message frequency is modified, the average value must be close to the average value under optimal conditions. The table shows a compilation of all the tests done for every model. The highlighted values represent the best results between

Table 4.1: Absolute value of group meeting point error and standard deviation. Each anticipation model shows the results by running the simulation without feedback (top value), and with feedback (bottom value), respectively. For example, for the 0.8 behavior with 6 steps/ message (S/M) and without anticipation, the TS and AUVs met 41.77 m away from the target point; with anticipation using no feedback and NN1, they met only 13.65 m away from the target point; and with anticipation using feedback and NN1, they met 14.95 m away from the target point.

Test (Model)			***No Anticipation***	***NN1***	***NN2***	***NN3***	***NN4***	***Fuzzy Logic***
SIMPLE	OnS	1 S/M	1.06(0.27)	1.06(0.28) 1.06(0.28)	1.06(0.28) 1.06(0.28)	1.06(0.27) 1.06(0.27)	1.06(0.28) 1.06(0.27)	1.06(0.28) 1.06(0.28)
		2 S/M	0.70(0.51)	0.89(0.26) 0.87(0.28)	1.01(0.28) 1.02(0.27)	1.02(0.27) 1.02(0.26)	1.00(0.27) 0.97(0.27)	1.16(0.28) 1.00(0.26)
		6 S/M	0.70(0.62)	0.75(0.28) 0.51(0.63)	0.99(0.27) 0.66(0.28)	1.00(0.27) 0.82(0.74)	0.93(0.28) 0.70(0.62)	1.21(0.29) 0.61(0.26)
	0.8S	1 S/M	10.98(0.18)	10.98(0.18) 10.98(0.18)	10.98(0.18) 10.98(0.18)	10.98(0.18) 10.98(0.18)	10.98(0.18) 10.98(0.18)	10.98(0.18) 10.18(0.28)
		2 S/M	28.30(0.27)	12.66(0.32) 12.66(0.19)	12.56(0.46) 12.27(0.19)	11.92(0.44) 11.95(0.44)	12.68(0.19) 12.43(0.46)	11.30(0.43) 11.30(0.43)
		6 S/M	41.77(0.26)	13.65(0.29) 14.98(0.28)	10.99(0.18) 14.90(0.31)	12.47(0.36) 13.74(0.06)	13.19(0.26) 15.65(0.42)	11.59(0.31) 14.05(0.25)
	1.2S	1 S/M	1.09(0.26)	1.09(0.26) 1.09(0.26)	1.09(0.27) 1.09(0.26)	1.09(0.26) 1.09(0.26)	1.09(0.26) 1.09(0.26)	1.09(0.26) 1.09(0.27)
		2 S/M	16.25(0.08)	2.87(0.27) 2.77(0.29)	3.83(0.28) 3.77(0.27)	3.53(0.27) 3.96(0.27)	3.81(0.27) 4.00(0.28)	1.63(0.25) 2.63(0.51)
		6 S/M	31.67(0.07)	4.46(0.26) 6.09(0.49)	5.07(0.27) 8.60(0.15)	5.02(0.39) 18.87(0.04)	4.93(0.26) 7.60(0.19)	2.47(0.32) 10.57(0.14)
COMPLEX	SwR	1 S/M	1.41(0.17)	1.40(0.17) 1.41(0.17)	1.40(0.72) 1.41(0.17)	1.41(0.17) 1.41(0.17)	1.41(0.17) 1.41(0.17)	1.41(0.17) 1.40(0.17)
		2 S/M	17.05(0.08)	0.53(0.19) 0.64(0.19)	0.07(0.16) 1.18(0.17)	0.20(0.16) 0.20(0.16)	1.08(0.22) 0.91(0.17)	1.89(0.17) 2.27(0.13)
		6 S/M	30.20(0.24)	0.54(0.33) 0.09(0.19)	0.96(0.23) 0.73(0.21)	0.55(0.28) 0.56(0.24)	1.68(0.18) 2.40(0.42)	2.50(0.37) 12.45(0.14)
	SwoR	1 S/M	5.96(1.95)	5.91(2.01) 5.98(2.04)	5.95(1.97) 5.96(2.03)	5.97(2.03) 5.93(1.98)	5.99(2.02) 5.84(2.13)	5.97(1.99) 5.95(2.06)
		2 S/M	30.69(0.20)	8.70(1.54) 8.60(0.19)	8.94(1.53) 8.14(1.81)	7.96(1.56) 7.98(1.57)	8.59(1.35) 8.85(1.61)	6.76(2.09) 6.76(0.13)
		6 S/M	45.14(0.39)	10.52(1.40) 12.63(0.19)	10.51(1.21) 14.07(0.53)	9.18(1.61) 10.99(1.09)	9.98(1.30) 12.76(1.98)	7.47(1.85) 12.45(0.14)
	Exp	1 S/M	2.99(0.80)	3.00(0.80) 2.95(0.80)	3.01(0.79) 2.99(0.78)	3.00(0.80) 3.03(0.81)	3.01(0.80) 2.98(0.79)	2.92(0.78) 3.01(0.68)
		2 S/M	11.64(0.42)	1.99(0.98) 2.49(0.92)	2.04(1.17) 2.60(0.67)	2.22(0.62) 3.38(0.49)	2.11(1.02) 2.52(0.80)	2.65(0.76) 2.99(0.61)
		6 S/M	18.72(0.04)	2.28(0.48) 2.88(0.27)	2.23(0.87) 3.73(0.73)	3.74(0.48) 5.42(0.43)	2.94(0.88) 17.34(0.14)	2.67(0.51) 5.47(0.14)
	T1	1 S/M	0.46(0.66)	0.45(0.66) 0.46(0.66)	0.46(0.66) 0.46(0.66)	0.46(0.66) 0.46(0.66)	0.45(0.66) 0.46(0.66)	0.46(0.66) 0.46(0.57)
		2 S/M	7.85(0.56)	0.87(0.59) 0.63(0.63)	0.78(0.29) 0.41(0.30)	1.10(0.52) 0.89(0.52)	0.81(0.64) 1.31(0.54)	1.05(0.49) 0.43(0.74)
		6 S/M	14.20(0.60)	0.19(0.44) 0.70(0.13)	0.35(0.36) 2.92(0.22)	2.15(0.33) 0.10(0.21)	2.63(0.73) 8.15(0.72)	1.83(0.40) 2.07(0.60)
	T2	1 S/M	2.07(0.39)	2.07(0.39) 2.07(0.39)	2.07(0.39) 2.07(0.39)	2.07(0.39) 2.07(0.39)	2.07(0.39) 2.07(0.39)	2.07(0.39) 2.07(0.37)
		2 S/M	3.44(0.25)	1.87(0.39) 1.88(0.42)	1.87(0.37) 2.04(0.39)	1.86(0.34) 2.04(0.36)	1.77(0.37) 2.00(0.36)	1.70(0.36) 2.14(0.45)
		6 S/M	4.14(0.42)	1.94(0.40) 2.52(0.44)	1.86(0.39) 2.34(0.33)	0.88(0.38) 2.24(0.26)	0.99(0.48) 2.75(0.29)	1.61(0.40) 2.27(0.38)

the five anticipation models for each test.

Table 4.1 shows that two models, NN1 and Fuzzy Logic, had better performance over the 32 different configurations used for the anticipation experiment. All five models showed a significant improvement over the experiments without anticipation when there were gaps of 2 and 6 steps between the messages. But the NN1 and Fuzzy logic models had better results on 12 and 9 of theses experiments, respectively. These two models showed the smallest error between the correct meeting point and the one obtained in the experiment. Although it also showed good results and the meeting point values were close to the best model (NN1), the NN4 model did not have the best results in any of the 32 different configurations used for the experiment.

The data in Table 4.1 suggest that the TS behavior can be categorized in two groups: simple and complex. The first 3 ship behaviors are relatively simple because the speed stays constant from the starting point until the meeting point. The other 5 TS behaviors are part of the complex behavior group because the TS changes its speed during its way to the meeting point. For the first group, the NN1 and Fuzzy Logic models each have the best results on 5 out of the 12 configurations. For the second group, the NN1 model is better than the other models on 7 of the 20 possible configurations, but the Fuzzy Logic model is best on only 4 of the 20. However, by looking only the complex behaviors, the NN3 model has a much better performance, moreover its performance is better than the Fuzzy Logic model by having better results in 5 out of 20 possible configurations.

On the other hand, the performance of the anticipation module, in general, cannot be classified as a better tool for simple problems or for complex problems. From Table 4.1, 2 pairs can be made according to their similar average errors: 0.8S Vs. SwoR and 1.2S Vs. SwR. Their averages, when there is no anticipation, is relatively close. By analyzing the values of 0.8S while the message frequency is decreasing (1 S/M $\rightarrow$ 6 S/M), the anticipation module is able to reduce the magnitude of the meeting

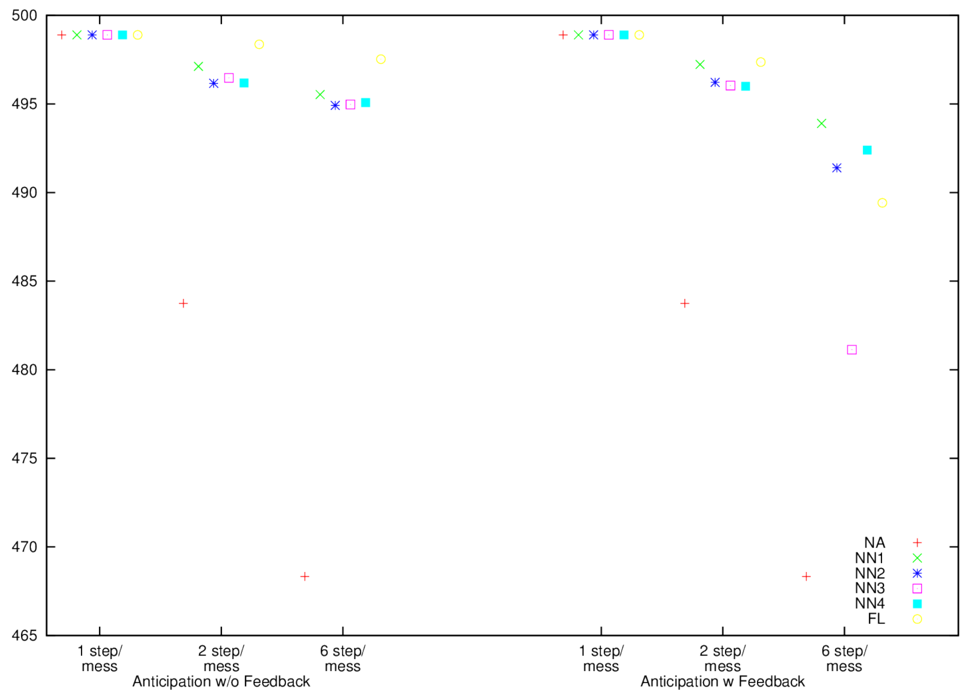


Figure 4.1: Global average of the meeting point results for the AUVs obtained with all anticipation models when the TS is using the 1.2S behavior.

point error better than in the SwoR behavior indicating that the anticipation does a better job with simple behaviors than with complex behaviors. However, by looking at the values from 1.2S and SwR behaviors, the analysis shows the opposite. Here, the SwR (complex) behavior was better on reducing the error magnitude than the 1.2S (simple) behavior.

Additionally, the anticipation model while using the Neural Network models also had good results with the TS behaviors, that were not included in the training data. The magnitude of the meeting point error was reduced just like when testing the TS behaviors inside the training data. The speed changes in the T1 and T2 behaviors include acceleration values that are not included in the other TS behaviors. This means that the Neural Network models are able to generalize these behaviors and recognize when the TS is speeding up or slowing down. These results also suggest that the Neural Network models are able to infer an anticipated message with similar values as the missing TS message.

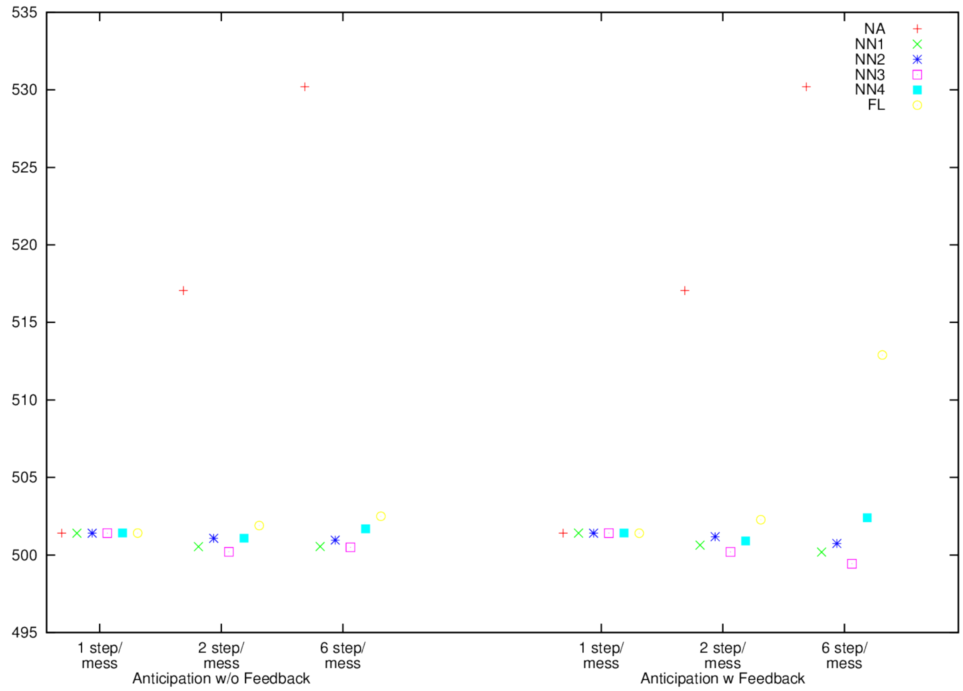


Figure 4.2: Global average of the meeting point results for the AUVs obtained with all anticipation models when the TS is using the SwR behavior.

Because this chapter is considered an extended part of Chapter 3, the data graphs described in this section follow the same format of the graphs presented in that chapter. Figures from 4.1 to 4.6 show the average meeting position of the group of AUVs with the TS when different TS behaviors and AUV anticipation models were tested. Figures from 4.7 to 4.10 show the average meeting point of each AUV with the TS by using the NN1 model and the Fuzzy Logic model which according to Table 4.1, were the models with better results. This last group of figures offers a better view of the AUVs while they try to maintain their alignment and meet the TS at the same time, even when anticipation is activated. Also, this last group of figures are focused on the SwR and the T1 behaviors of the TS. These 2 behavior will allow us to analyze not only the behavior of each AUV during complex TS behaviors but also when the behavior is not part of the training data of the Neural Network.

Figure 4.1 is presented and explained in this chapter, although the description for the 1.2S behavior is presented in Chapter 3. Due to size limitations set by the

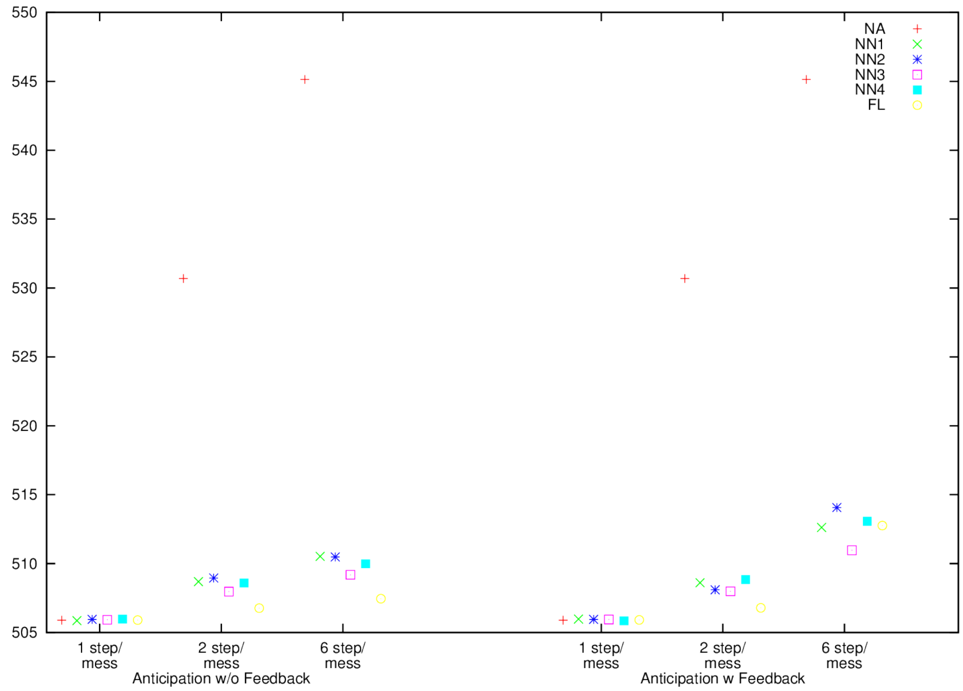


Figure 4.3: Global average of the meeting point results for the AUVs obtained with all anticipation models when the TS is using the SwoR behavior.

MTS/IEEE Oceans'13 Conference rules, the graph for the 1.2 speed behavior was not included in the official version of the paper, but its data was well documented. Figure 4.1 shows that all of the anticipation models were able to reduce the distance to the meeting point when the messages were infrequent. NN3 was the only model that had a poor performance, but only when the experiment was run with anticipation with feedback and 6 cycles/message. Figure 4.1 shows that this result is separated from the other results which show a common behavior.

On the other hand, the anticipation models showed clustered results while the TS was running under the SwR and SwoR behaviors (Figures 4.2 and 4.3, respectively). Here, all the results are meeting very close to reference meeting point. For both behaviors, the error increases very fast when there is no anticipation and the frequency of messages is reduced. The anticipation modules reduce the error and their results are clustered inside a defined area, less than 5 meters away from the optimal meeting point, unlike the results shown previously in Figure 4.1. Figure 4.4 shows some results

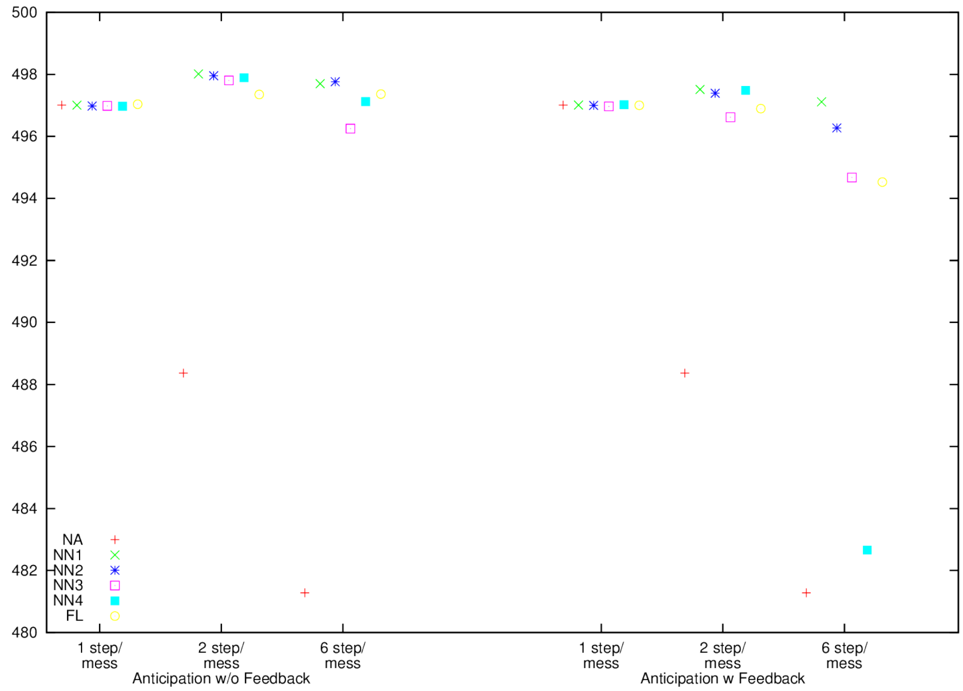


Figure 4.4: Global average of the meeting point results for the AUVs obtained with all anticipation models when the TS is using the experimental behavior.

when the TS is running under the experimental behavior and these results are close to the reference meeting point as well, but some of them have a small dispersion. This dispersion is small but, by comparing this figure with Figures 4.2 and 4.3, it is evident. It is important to remember that the TS increments its speed a lot at the end of the run during this behavior, which can have a large influence on the results, especially when the message frequency starts to drop.

The results shown in Figures 4.5 and 4.6 are for the Test1 and Test2 behaviors. For the Neural Network anticipation models, these behaviors are not part of the training data. These behaviors were used to test the adaptability of the Neural Networks and to confirm the effectiveness of all anticipation models, including the Fuzzy Logic model. These Figures show that the Neural Networks are able to generate a similar message to the missing TS message to control the AUVs, even when the TS behavior was not included in the training data. While using anticipation, the AUVs were able to infer the speed changes for different acceleration rates. During the simulation, all

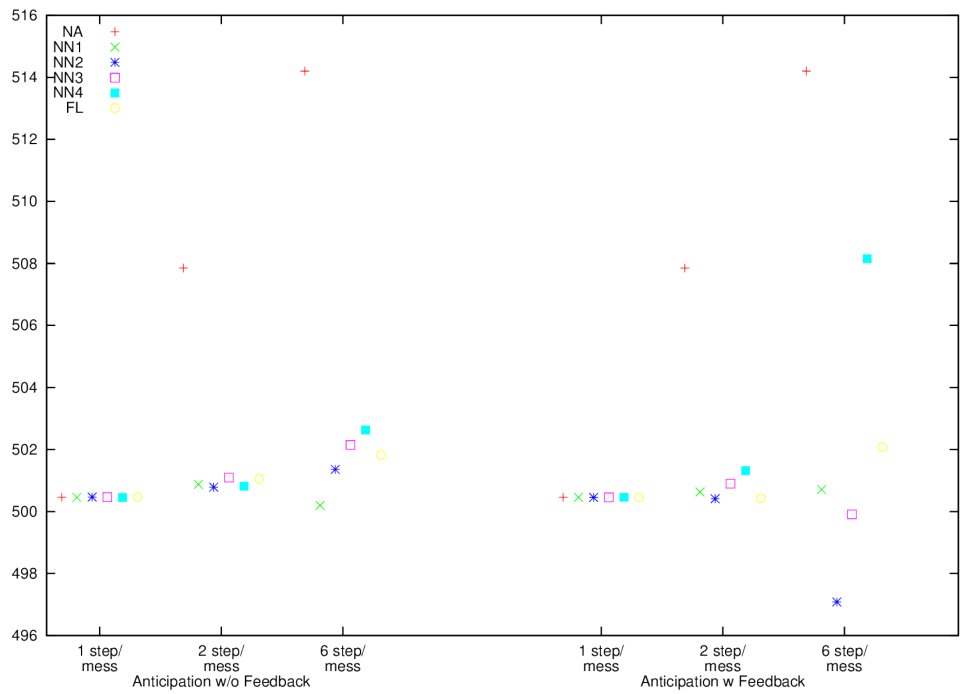


Figure 4.5: Global average of the meeting point results for the AUVs obtained with all anticipation models when the TS is using the T1 behavior.

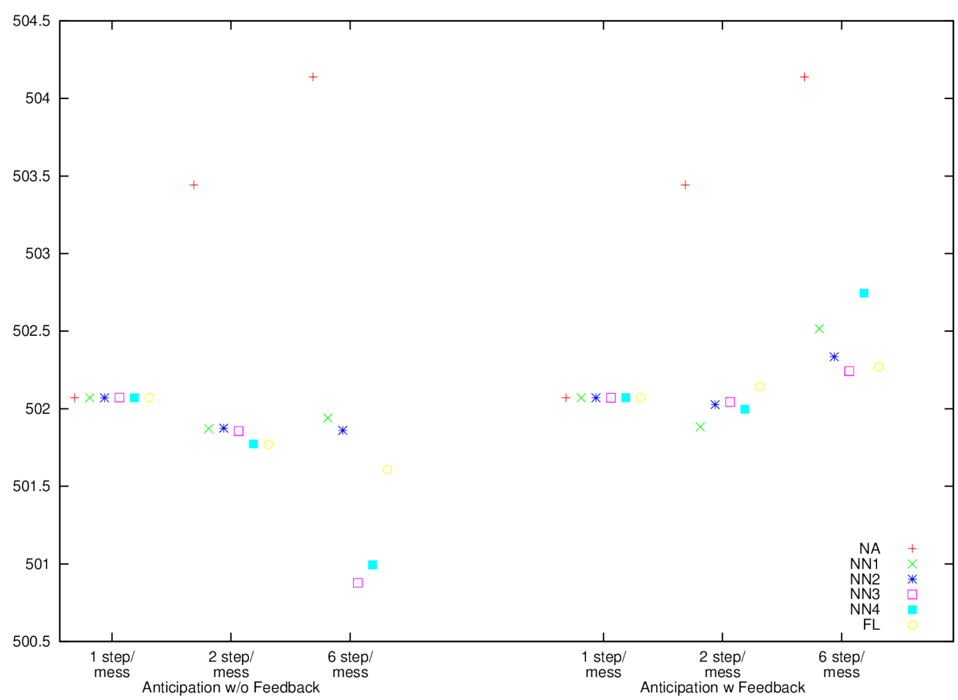


Figure 4.6: Global average of the meeting point results for the AUVs obtained with all anticipation models when the TS is using the T2 behavior.

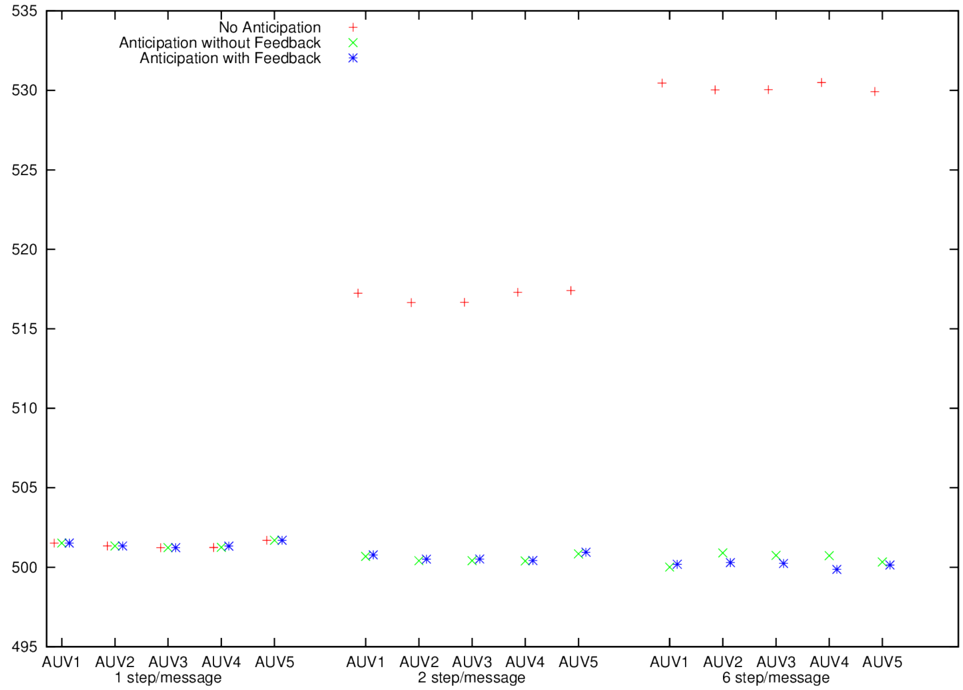


Figure 4.7: Average and quartile results for the meeting point by AUV, when using a neural network with 10 inputs and 5 neurons in the hidden layer when the TS is using the SwR behavior.

the anticipation models, including the Fuzzy Logic model, were able to detect when the TS was supposed to slow down or speed up. Not all of the Neural Networks were completely successful in all cases, but they still were able to reduce the error of the meeting point considerably.

Figure 4.7 and 4.8 show the average position for each AUV when the TS behavior is set to SwR and T1, respectively. Each AUV is using the NN1 model as the anticipation model. In Figure 4.7, the results show that the AUVs are able to maintain formation even when the message frequency decreases which means that the anticipation module is creating a useful message for the AUV speed controller. The anticipation module helps to reduce the meeting point error considerably for the entire group of AUVs; in this case the error reduction is around 28 meters. Also, Figure 4.8 also shows an error reduction for the T1 behavior when anticipation is activated. When the message frequency is low, this error seems to be higher than the

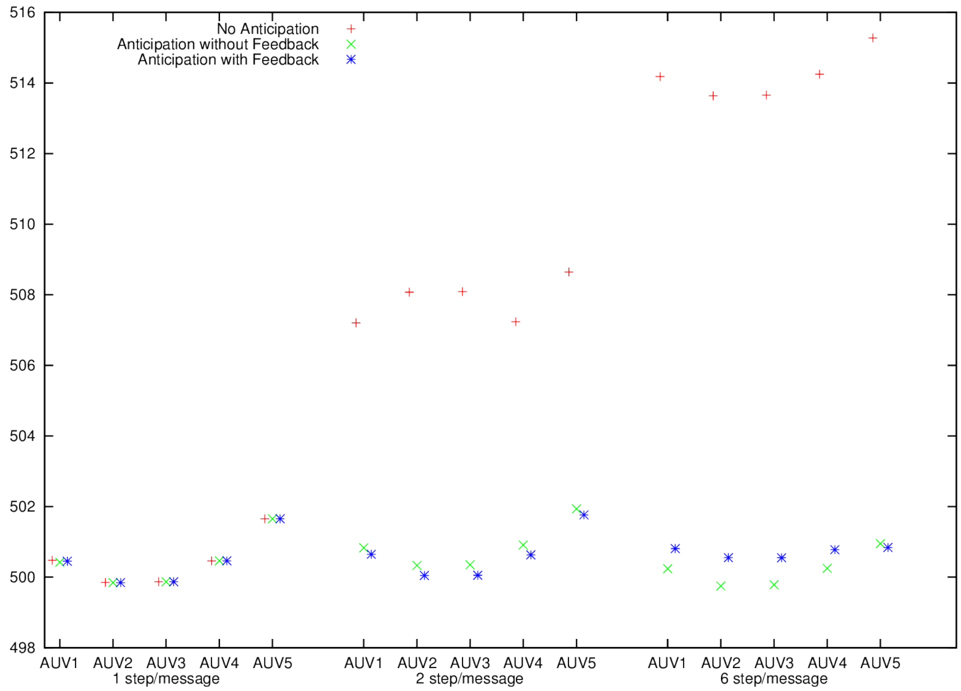


Figure 4.8: Average and quartile results for the meeting point by AUV, when using a neural network with 10 inputs and 5 neurons in the hidden layer when the TS is using the T1 behavior.

error shown in Figure 4.7. This suggests that the Neural Network is able to solve the problem for T1 even when this TS behavior is not part of the training data. For the T1 behavior, the results show that the AUVs are not able to keep formation when no anticipation was used. But they are able go back in track and maintain a close formation when the anticipation module is activated.

Figures 4.9 and 4.10 show the average position for each AUV when the TS behavior is set to the SwR and T1 behaviors, respectively. Compared to the configuration that was set for Figures 4.7 and 4.8, the only difference is that the AUVs in 4.9 and 4.10 are using the Fuzzy Logic model as the anticipation model. In Figure 4.9, it can be seen that the anticipation module had some difficulty inferring the correct messages when the module was using feedback and the message frequency was low, otherwise it was able to reduce the error in each AUV considerably. However, the anticipation module was able to help the group of AUVs maintain formation even when the error

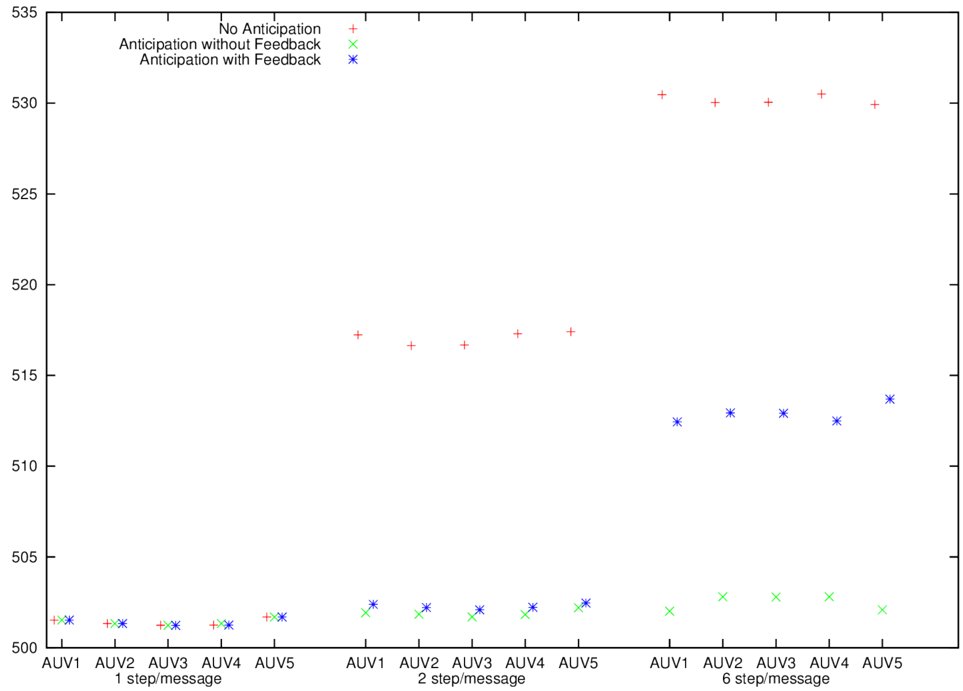


Figure 4.9: Individual average of the meeting point results obtained using Fuzzy Logic when the TS is using the SwR behavior.

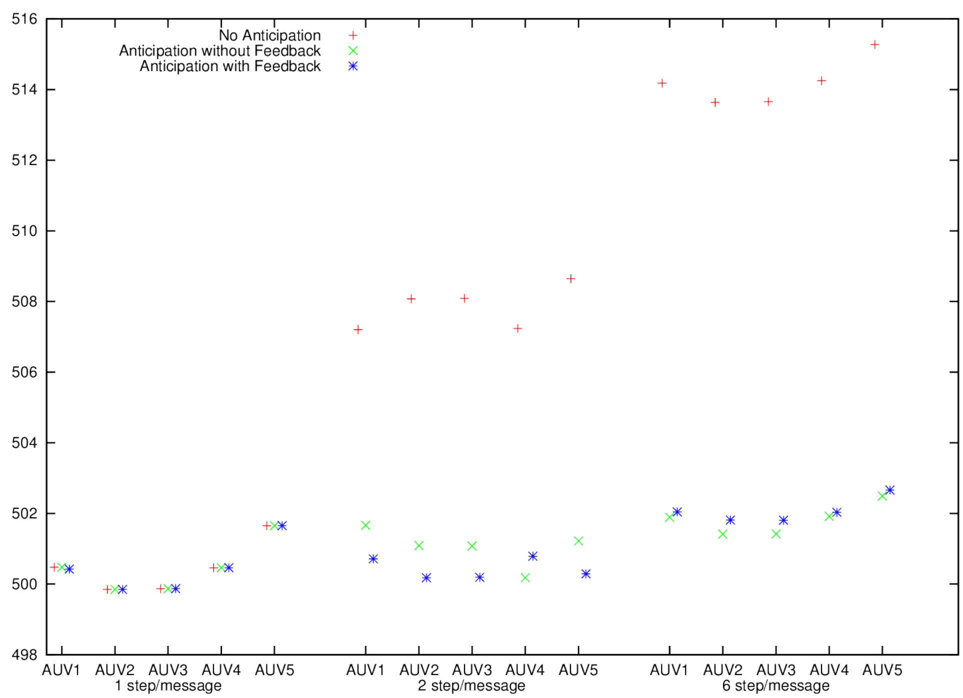


Figure 4.10: Individual average of the meeting point results obtained using Fuzzy Logic when the TS is using the T1 behavior.

Table 4.2: Percentage error correction of the meeting point for each anticipation model with a message frequency of 6S/M. For example, NN2 was able to reduce the meeting point error during the Exp behavior by 95.17 percent of the result with No Anticipation.

Test \ Model		***NN1***	***NN2***	***NN3***	***NN4***	***FL***
Simple	0.8S	91.33	99.97	95.16	92.82	98.02
	1.2S	88.98	86.98	87.15	87.44	95.49
Complex	SwR	96.98	98.44	97.01	99.06	96.21
	SwoR	88.36	88.39	91.78	89.74	96.15
	Exp	95.49	95.17	95.23	99.68	97.97
	T1	98.03	99.20	87.70	84.21	90.03
	T2	93.72	89.85	42.51	47.83	77.78
Average		93.27	94.00	85.22	85.83	93.09

reduction was not fully successful; see Figure 4.10. In the best case, the anticipation module was able to reduce the meeting point error by about 28 meters. On the other hand, the results for the T1 behavior in Figure 4.10 are more clustered than in Figure 4.9. Here, the anticipation module was able to reduce the meeting point error by about 12 meters. And as in the previous figure, the AUVs were able to maintain formation with the help of the anticipation module.

To have a general view of the performance of all five anticipation models, the average error reduction was calculated. This value is the percentage of the error reduction based on the obtained error when no anticipation model was used. For this case, we use the most extreme conditions in the simulation which is 6S/M. First two errors are calculated: the error using no anticipation and the error using anticipation.

$E_{\text{NA}} = E_{NA-6S/M} - E_{NA-1S/M}$

$E_{\text{AM}} = E_{NA-6S/M} - E_{AM-1S/M}$

Using this values, the error delta and the percentage error is calculated:

$\Delta E_{\text{AM}} = E_{\text{NA}} - E_{\text{AM}}$

$P\,(\text{AM}) = \frac{\Delta E_{\text{AM}}}{E_{\text{NA}}}$

Based on the results of Table 4.2, NN1, NN2 and FL models hava an average percentage of error correction greater than 93%. The NN1 model shows a more stable performance with all TS behaviors. On the other hand, the Fuzzy Logic model has better except for the T2 behavior which, in general, was a difficult task for all anticipation models, excluding NN1.

4.3 Conclusions

Table 4.1 shows that the anticipation models used on each AUV were able to correct the position error on the meeting point with the TS when the messages were infrequent. Some models were more successful than others for specific cases. NN1 had, in general, performed best both the simple and the complex behaviors followed by the Fuzzy Logic model when measured by the number of results with the lower meeting point error. By looking at Table 4.2, NN2 shows a better performance based on the percentage of error reduction, although NN1 and the Fuzzy Logic models have a percentage greater than 93% as well. On the other hand, the NN2 and NN3 models had better results when they were trying to solve the complex models.

The data shows that all 5 anticipation strategies are able to considerably reduce the error generated when the message frequency is low. The messages that the anticipation module generates with each model need to be similar to the TS messages that were supposed to be received by the AUVs in order to generate a similar behavior as when the AUVs do receive a message every cycle, otherwise the speed controller in the AUVs would show different performances for the test with no anticipation and messages every cycle and the test with anticipation and gaps between messages. These results show that the AUVs try to mimic the behavior of anticipation by using learned behavior and a limited information to generate an hypothetical yet helpful answer. However, it cannot be directly determined if the anticipation models perform, in general, better on simple or complex behaviors. for this type of models, this assumption can be vague because, at the end, the performance will depend on the selected training data for the neural networks or the distribution of the fuzzy sets and fuzzy rules. In this case, there was pair of TS behavior (0.8S and SwoR) that would suggest that the anticipation module would work better on simple behaviors. But there was also an additional pair (1.2S and SwR) that would suggest that the anticipation model would work better on complex behaviors.

By analyzing all anticipation models, especially the ones using Neural Networks, it can be concluded that these models, after a successful training phase, are able to generalize and identify when the TS is slowing down or speeding up and they are able to calculate a rough approximation for the magnitude. It is important to point out that the AUVs do this only by using a small sample from the previous message history (2 or 5 previous messages depending on the model).

Chapter 5: Anticipation Strategy for Speed Control in a Commodity Off the Shelf (COTS) Robot Fleet

5.1 Introduction

In the previous chapters, it has been demonstrated in simulation that the anticipation models proposed for a fleet of AUVs have good performance and are able to overcome disrupted and noncontinuous communications to match the speed of the AUVs with the TS speed. This chapter shows that these anticipation models can also be used with physical robots recreating the same mission of a TS meeting an AUV fleet. During the simulation experiments, there were factors, like hardware limitations or noise generated by sensors, that were not part of the environment and the evaluation of the data. In this set of experiments, the anticipation module is tested in a more complex environment. Here, using real robots introduces additional variables that can add noise to the system. The goal is to determine whether the anticipation module is able to generate appropriate messages to overcome these problems.

Other research projects have shown that the communication between a group of robots can be essential for the stability and success of a mission. In [5], the authors emphasize that communication between the robotic agents is a key factor in completing a cooperative task. In that case, the group of robots established a short-ranged wireless link in order to interact between each other and establish a convoy-like formation. The authors in [3] point out that the communication between mobile robots is very important for cooperation, although it is not necessary to use a conventional wired or wireless communication; robots can also communicate by using the environment (stigmergy-like strategies) or by using build-in sensors (cameras, bump sensors, sonars, etc.). If communication is a main factor in coordinating cooperative autonomous robots, the robots must be prepared in case communications are broken or disrupted. This chapter shows a similar approach to the one described in Chapters 3 and 4 for a simulated environment, only this time it is applied in a real-world environment where additional factors might be involved in the performance of the robots.

This chapter is organized as follows: Section 5.2 presents a detailed description of the robot hardware, image processing and communication module which the robot uses as additional tools for its functionality. Section 5.3 presents the different Fuzzy Logic controllers that were used to control the robot and to generate the messages. Section 5.4, presents 3 behaviors and 3 communication situations used as the experimental conditions to test the system. Finally, section 5.5 presents the conclusions.

5.2 Experimental Platform

For this experiment, 2 robots with similar hardware features were used. One of the robots can be selected by the user as the Target Robot (TR), the equivalent to the TS in Chapter 3. The second robot can be selected as the Autonomous Robot (AR), the equivalent to an AUV in the simulation. For these experiments, there is only a single AUV robot, not five. The next subsections describes the hardware and the main processes that the robots use for their functionality.

5.2.1 Hardware

The robot hardware consists of 3 main parts: the body, the “spinal chord” and the “brain”. The three parts of the mobile robot can be appreciated as a whole structure in Figure 5.1.

- Body: it consists of an aluminum chassis with two servo motors that control the speed and steering of the robot. This robot has some mechanical limitations for steering, the maximum it can turn is 33 degrees. For these experiments the steering motors is only used to get the robot back on track to the goal by using

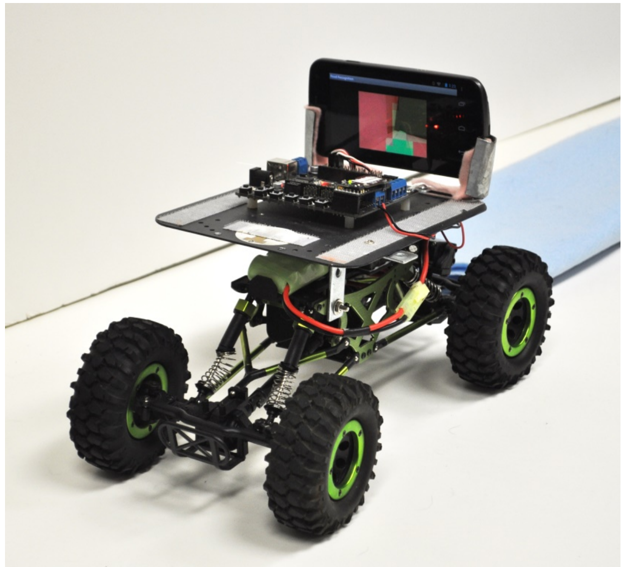


Figure 5.1: Mobile Robot structure that includes the body, the spinal chord and the brain.

a control signal to turn left or right that tries to maintain the detected object centered in the screen of the smartphone camera. The second motor is in charge of the speed and it uses speed values between stop and full forward; there is no backward command for these experiments.

- Spinal Chord: it represents the connection between the brains and the body of the robot. Here, the spinal cord of the robot is an Arduino Romeo board. This board receives signals sent from the "brain" and converts them to the values that the body will use to control the motors.

- Brain: it consists of a smartphone HTC Desire running Android 2.3.4. It is in charge of the heavy processing by the robot. The brain runs high-demand processes like image processing and message generation. It also runs the GUI that allows the user to select different options that the robots can implement during the experiments.

Figure 5.1 shows that the smartphone can display on the screen what the camera is capturing and what the program is understanding from the image. At the moment a

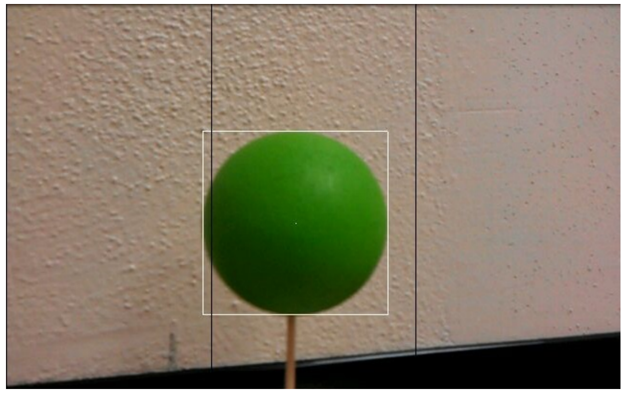


Figure 5.2: Screen capture of the object detection method used in the smarthphone.

desired object is detected, it is marked on the screen with a white square surrounding it (Figure 5.2). This gives the user a better perspective of what the robot is seeing and detecting and allows him to have a better understanding of the robot's behavior.

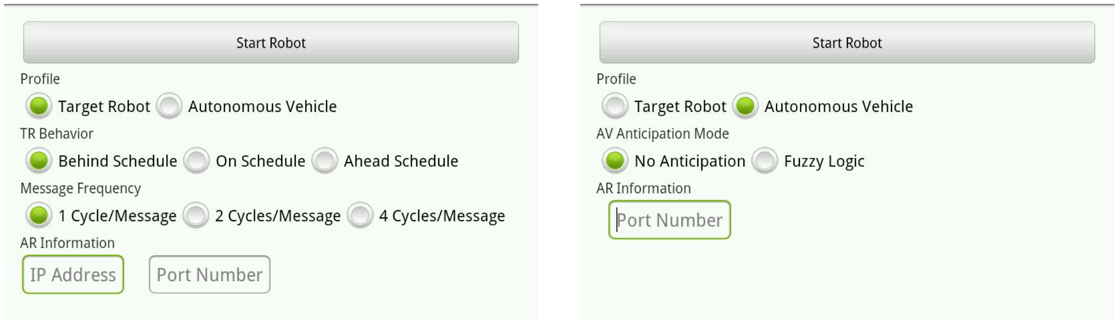
The application installed in the smartphone integrates the options to configure both the TS and AR behaviors. In the main screen of the app (Figure 5.3a), the user can select the following options for the TS:

- TS behavior: this option allows the user to set the behavior of the TS between 3 different choices (Behind Schedule, On Schedule and Ahead of Schedule).

- Message Frequency: this option sets the number of steps that the TS would wait to send the next message to the AR. The unit used for this option is represented in Steps/Message (S/M). A step represents an iteration of the program in the robot in which frame captured by the camera is analyzed followed by the derived decision. Three choices are available (1 S/M, 2 S/M, 4 S/M).

- AR Information: this option lets the user enter information about the server that will be running on the AR side. This information allows the TS to establish a TCP communication with the AR. Here, the TS works as a TCP client. The required information is the server IP address and port number.

The other options in the main screen for the AR are (Figure 5.3b):



(a) TR menu interface (b) AR menu interface

Figure 5.3: GUI for anticipation app

- AR Anticipation Mode: this option allows the user to choose between turning off the anticipation module or turning on the anticipation module with the Fuzzy Logic model.

- AR Information: this options lets the user introduce the information for the server running on the AR side. The AR will always run as the server and it only needs the user to set the port number.

5.2.2 Image Processing

The camera in the smartphone captures an image, which is passed to a method running in the smartphone for processing. This process is done using the OpenCV image libraries. First, a color detection method identifies whether there is a green object in the image which will be marked in the camera screen with a square to tell the user that it has been identified (Figure 5.2). If a green object is detected, the method calculates the number of pixels in the green object and then it calculates the x-axis of the object to obtain the diameter of the object. The method returns this value which is passed to the next method to calculate the distance from the robot to the detected object. In this case, we are using a ball with a diameter of approximately 2.5". With this ball size, the equation to calculate the distance to the ball, according to the number of pixels detected in the image, is distance $= \frac{45}{\text{diameter}^2}$. where the diameter is measured in pixels and the distance is measured in meters. This formula was generated via empirical testing.

5.2.3 Communication

The robot uses two types of wireless connections to transfer information. The smartphone and the Arduino board use a Bluetooth connection to send the driving commands from the phone to the Arduino. For the communication between robots, the smartphones use a WiFi connection to send messages regarding progress from the TR to the AR. Both modules are created as a thread in order to be called while the image processing is working. The connection of both modules is created by using sockets, which requires setting up a server and a client on each end. The message type for both modules is byte array and each module has its own method to encapsulate the values to be sent.

Each communication module (Bluetooth and WiFi) has an independent activation in order to send a message to its destination. The Bluetooth module is activated at the end of every cycle when the image processing module has finished analyzing the picture and the resulting value is used to calculate the values sent to the motors. Based on empirical data, the mean time for the robot to send a message by Bluetooth is 100.59 ms with a standard deviation of 31.88 ms. For the WiFi communication, we chose the messages to be sent from the TS to the AR at fixed intervals in order to have a controlled environment during the experiments. In this case, a value of 150 ms (greater than 100.59 + 31.88 ms) was chosen in order to guarantee that a new message was sent from the TS only after obtaining the final result from the image processing and message generator.

- Bluetooth: the messages sent via Bluetooth have a special format in order

Table 5.1: Bluetooth message structure

#	T	[56 - 120]	P	[45 - 129]	;

to check for corrupted messages on the receiving side (Arduino board). Each message consists of a 6 byte array and its format is shown in Table 5.1. This format allows the receiving side to check easily for transmission errors. The message starts with a header represented by the # symbol. The header is followed by the character *P* which indicates that the next byte contains the value for the steering motor. Normally, the range for the steering value goes from 0 to 180. But, in this case, it is limited to a range between 56 and 120 because of the steering limitations of the robot. the fourth byte uses the character *P* which indicates that the next byte contains the value for the speed motor. Here, the range for the speed motor is also limited between 45 and 129 because those are the maximum values in which the robot reaches its speed while driving backwards and forward, respectively. For these experiments, only half of this range is used because we are only interested in using the stop and forward actions. The fifth and last byte is the *;* character which represents the end of the message.

- WiFi: this module captures the message created by the TR and sends it to the AR. This message contains a double value which is wrapped in a byte array using the ByteBuffer library from Java. The AR uses the same library to unwrap the message from the byte array and transform it into a double again.

5.3 Methods

The application and the experiments were set up to work with values that the user inserts at the initial screen. The TS and AR profiles are selected for each robot. Also, the parameters TS behavior, TS message frequency, and AR behavior are selected. The TS behavior represents the speed of the TS during the test and the TS message frequency represents the number of cycles that the TS will wait before sending a message to the AR, just as it was done in Chapter 3. The AR behavior module allows the user to activate or deactivate the anticipation module for the tests. Additionally, the robots use 3 modules to run the Fuzzy Logic controller. These modules are: Message Generator, Speed Increment Calculator and Anticipated Message Estimator.

The initial process for both robots is to capture the image and calculate the distance from the robot to the detected ball on screen. If the ball is not detected by the camera, the main process is halted and waits for the next image to be processed. With the distance value, each robot calculates a value between -1 and 1 with a fuzzy logic controller inside a module called the Message Generator. This value represents the relative status of the robot to the green ball. For example, negative values mean that the robot is behind schedule, zero means that it is on schedule and positive values mean that it is ahead of schedule. The TR sends this value to the AR which compares this value to its own status message generated by another Message Generator module inside the AR. An additional module, denominated as Speed Increment Calculator module, uses this difference and its output is used then to adjust the motor speed, so the robot will arrive at the goal at the scheduled time. It is important to point out that the robots have similar but not identical parameters for their motors. This means that the same value to the drive motor will result in different speeds depending on which robot is used and on battery charge. For this reason, the fuzzy logic controller that calculates the speed adjustment is used by the TS as well. By using the value that user selects for the TS behavior, the TS can control its own speed based on the default speed control model that was previously generated by sampling of one robots.

The Message Generator and the Speed Increment Calculator Modules are the basic structure of the system and if the anticipation behavior on the AR is deactivated the third and final module, the Anticipated Message Estimator module, is not used. This last module is only used by the AR and it is called only when anticipation is selected before the test starts. The AR has a double-type array of 2 values that is used as a queue. This queue stores a TR message at the moment the AR receives it by erasing the oldest message. When the AR detects that a message did not arrive while anticipation is on, it calls the Anticipated Message Estimator module. This module uses another Fuzzy Logic controller that reads the message queue (without deleting its content) and generates an anticipated message to replace the missing TR message. This new message is intended to be an estimation of what the missing message was supposed to be. Then, the AR uses the anticipated message in the Speed Increment Calculator module to generate a new adjustment value for the speed motor.

5.3.1 Message Generator Module

First, the message generator calculates the input to the Fuzzy Logic controller by using the current speed of the robot and its current distance to ball. The following equations were created by sampling one robot's parameters while its speed was being changed.

Normally, the robot uses a variable in a range of [0.5 - 1] to control the forward speed motor. By using this equation, this variable can be converted to meters/seconds:

$speed = 0.8443 \times motorspeed - 0.3531$

Then, the distance obtained by image processing module has to be recalculated in order to include two additional factors. The robot includes a flag to stop before reaching the ball in order to avoid running over it and an estimation error is included that changes according to the current speed of the robot. This last variable was added because it was detected that when the robot goes faster, its reaction time for stopping increases. This happens because the smartphone also includes a processing delay in analyzing the image. To compensate for the delay caused by the robot's speed, the following equation was used:

$\text{error}_{dist} = 0.0865 \times \text{motorspeed} - 0.0489$

The estimated distance left to reach the ball is calculated by subtracting the error and the stop threshold from the distance obtained by the image processing module. The stop threshold ($\text{Dist}_{\text{THRESHOLD}}$) was set to a value of 0.35 meters.

$\text{dist}_{est} = \text{dist}_{camera} - (\text{Dist}_{\text{THRESHOLD}} + \text{error}_{dist})$

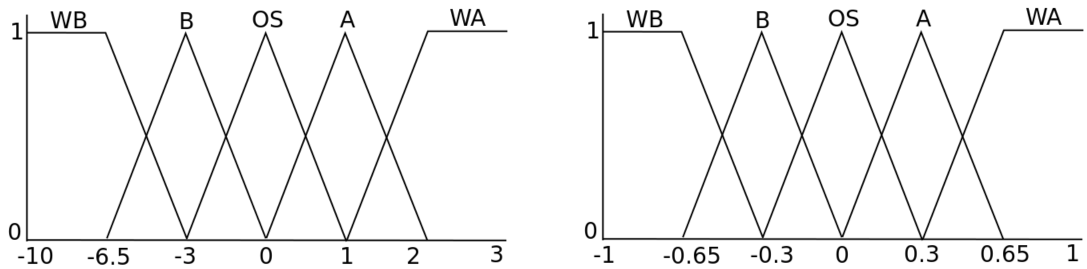
After calculating the estimated distance, the robot calculates the estimated time it has left to reach the ball by using its current speed. The estimated time is converted to milliseconds because those are the units that the smartphone use with its inner clock.

$\text{Time}_{est} = (\text{dist}_{est}/\text{speed}) * 1000$

The last step before using the fuzzy logic controller is to calculate the scheduled time. This variable represents, in milliseconds, how much time the robot has left to reach the goal. It uses a variable defined as GoalTime, which is a constant that represents the time the robot takes to reach the goal at its regular or "on schedule" speed. In these experiments GoalTime is equal to 7074 ms based on a starting distance of 3 meters to the threshold point. An additional variable is CurrentTime, which is the current time. This variable is calculated by using the clock of the smartphone. The final equation is:

$\text{Time}_{sched} = \text{GoalTime} - \text{CurrentTime} - \text{Time}_{est}$

Time_{sched} is used as the input to the Fuzzy Logic controller which uses the following fuzzy sets: WayBehind (WB), Behind (B), OnSchedule (OS), Ahead (A) and WayAhead (WA). The output of the Fuzzy Logic controller uses the same names for its fuzzy sets but their limit values are different from the input fuzzy sets. Both input and output fuzzy set are shown in Figure 5.4.



(a) Scheduled Time Input (b) Message Output

Figure 5.4: Fuzzy Sets that describes the current value of the input and output of the Fuzzy Logic controller in the Message Generator module. The fuzzy sets are: Way Behind (WB), Behind (B), On Schedule (OS), Ahead (A), Way Ahead (WA)

Table 5.2: Fuzzy Rules for Message Generator

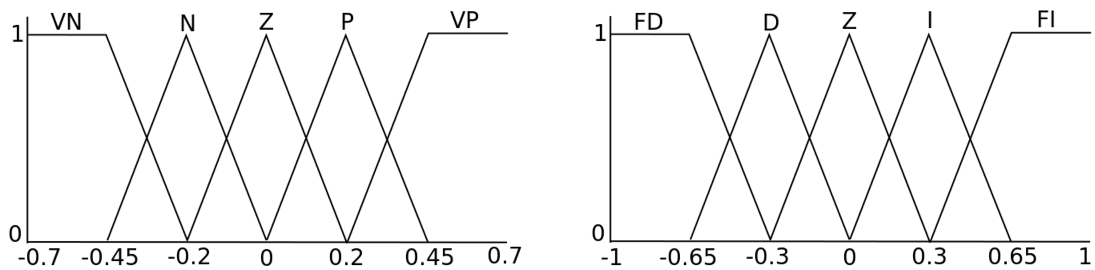
Scheduled Time				
WB	**B**	**OS**	**A**	**WA**
WB	B	ON	A	WA

The Fuzzy Logic Controller evaluates the input and output sets by using the rules described in 5.2. The function of the Fuzzy Logic Controller and these rules is to set the TR and AR messages inside a normalized region between -1 and 1. By using this range, an equivalent distribution of the TR progress can be obtained for negative behaviors (Way Behind and Behind) and for positive behaviors (Ahead and Way Ahead).

5.3.2 Speed Increment Calculator Module

This module uses the difference between two messages to generate the input to the Fuzzy Logic Controller. In general, the first input represents the reference message and the second input is the message obtained from the Message Generator Module. The input to the Fuzzy Logic controller is represented as: $\text{diff}_{\text{messages}} = \text{message}_{\text{Reference}} - \text{message}_{\text{MessGen}}$

Each robot profile uses a different reference message. The TR uses the selected TR behavior in the menu as a reference message: Behind Schedule (-0.5), On Schedule



(a) Message Difference Input

(b) Speed Increment Output

Figure 5.5: Fuzzy Sets that describes the current value of the input and output of the Fuzzy Logic controller in the Speed Increment Calculator module. The input fuzzy sets are: VeryNegative (VN), Negative (N), Zero (Z), Positive (P) and VeryPositive (VP). The output fuzzy sets are: FastDecrease (FD), Decrease (D), Zero (Z), Increase (I) and FastIncrease (FI).

(0), Ahead of Schedule (0.5). This reference message gives a more steady behavior on the TR under the desired parameters. For example, it is likely that the On Schedule behavior can have different effects while is being used on two different robots working as TR. This attempt is to correct hardware differences in the TR that can alter the selected speed

On the other hand, the AR uses the message received from the TR as the first message. In this case, the TR message works as the reference message and the goal of the AR is to try to match the TR speed by interpreting the progress of the TR described in the message.

The input variable for the Fuzzy Logic Controller of the Speed Increment Calculator is mapped to the following fuzzy sets: VeryNegative (VN), Negative (N), Zero (Z), Positive (P) and VeryPositive (VP). The output for the Fuzzy Logic Controller uses the following fuzzy sets: FastDecrease (FD), Decrease (D), Zero (Z), Increase (I) and FastIncrease (FI). Both input and output fuzzy sets are shown in Figure 5.5.

The Fuzzy Logic Controller evaluates the input and output sets by using the rules described in 5.3. The result of this module is used to modified the current speed of the robots:

$$\text{motorspeed} = \text{motorspeed} + 0.05 \times \text{SpeedIncrement}$$

Table 5.3: Fuzzy Rules for Speed Increment Calculator

Message Difference				
VN	**N**	**Z**	**P**	**VP**
FD	D	Z	I	FI

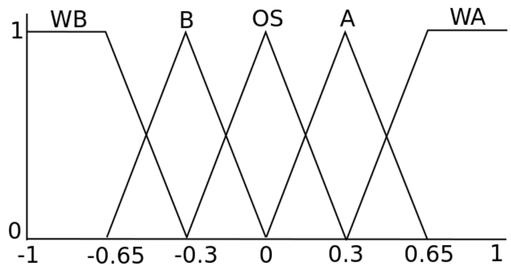
5.3.3 Anticipated Message Estimator Module

This last module is used only by the AR when the anticipation option is activated. In order to have a controlled and synchronized environment, the TR and the AR send and receive messages every 150 ms, respectively. Once the AR starts receiving messages from the TR, it saves the incoming message in a double-type array of size 2 that behaves as a queue. The size of this array represents the memory of the anticipation module. With this format, the anticipation module will only be able to remember and use the past 2 TR message. This configuration helps to prove that it is unnecessary for the anticipation module to save and use the entire history of TR messages to generate an anticipated message. When a new message arrives, the values are shifted and the message at the end of the array is deleted. When a message is received by the AR, it runs the previous modules normally. But when the AR detects that no new message has arrived, it runs the Anticipated Message Estimator to generate a message with the same format as the TR message by using the message history array as temporary memory about the previous behavior of the TR.

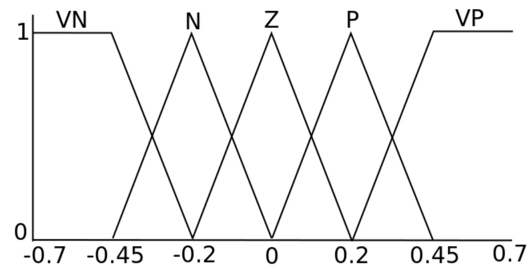
The Fuzzy Logic controller in this module is slightly different from the ones used in the previous modules. This Fuzzy Logic controller uses 2 input variables instead of one. Here, the two inputs are the most recent message in the array and the difference or error between the two messages in the array.

$\mathrm{diff}_{\mathrm{TRmessages}} = \mathrm{NewMessage} - \mathrm{OldMessage}$

The input variable that represents the most recent message uses the the following



(a) New Message Input



(b) Message Difference Output

Figure 5.6: Fuzzy Sets that describes the current value of the inputs for the Fuzzy Logic controller in the Anticipated Message Estimator module. The input fuzzy sets are: WayBehind (WB), Behind (B), OnSchedule (OS), Ahead (A) and WayAhead (WA). The output fuzzy sets are: VeryNegative (VN), Negative (N), Zero (Z), Positive (P) and VeryPositive (VP).

Table 5.4: General fuzzy rules for the Anticipated Message Estimator

Error \ **Message**	*WB*	*B*	*OS*	*A*	*WA*
VN	WB	WB	B	OS	OS
N	WB	WB	B	OS	A
Z	WB	B	OS	A	WA
P	B	OS	A	WA	WA
VP	OS	OS	A	WA	WA

fuzzy sets: WayBehind (WB), Behind (B), OnSchedule (OS), Ahead (A) and WayAhead (WA). The input variable that represents the difference between the messages uses the following fuzzy sets: VeryNegative (VN), Negative (N), Zero (Z), Positive (P) and VeryPositive (VP). Both fuzzy input sets are shown in Figure 5.6.

The output of the Fuzzy Logic controller is a message with the same format that the TR sends. This means it has the same format that the Fuzzy Logic controller uses as an input. Thus, the fuzzy set for the output must have the same format as the one used for the input, as in Figure 5.6a. The Fuzzy Logic Controller evaluates the two inputs by using the rules described in 5.4 to calculate the appropriate output value.

This method allows the AR to infer what would have been the missing TR message

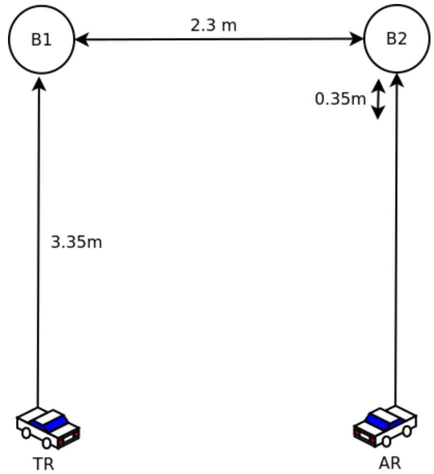


Figure 5.7: Testing arena designed to test the anticipation problem using two real robots.

and use it to update its own speed.

5.4 Results

In order to test the robots on the anticipation problem, a similar environment to the one described in Chapter 3 was used. In this case, an area 3.5 x 2.5 meters was used where two mobile robots have to drive towards two green balls (Figure 5.7). An independent target (a green ball) was selected for each robot in order to avoid collisions between the robots when they reach the green ball. One robot acts as the Target Ship (TS) or Target Robot (TR). It sends progress messages to the second robot every 150 milliseconds. The second robot acts as the AUV or the Autonomous Robot (AR). This robot receives the message from the TR and it tries to adapt its speed to match its progress with the TR's progress. The robots have a warning flag at 0.35 meters from the target that warns them to stop and avoid running over the green balls. A graphic illustration of this testing arena is shown in Figure 5.7.

For these experiments, all the features of the TR and AR were combined and tested. For every combination, 6 samples were taken, which were used to generate the statistical data. For the complete experiment, a total of 108 samples were taken.

Table 5.5: Average meeting point error and standard deviation while using No Anticipation (NA) and the Fuzzy Logic model (FL) as anticipation. The results represents the final position of the AR when the TS reached the target. The error value is based on the flag that the robots detect to stop 35 centimeters before reaching the ball. For example, for the BS behavior with 4 time steps per message (4 S/M), the AR was 120 cm from the stopping point when the TR reached the target.

Test \ **Model**		***NA***	***FL***
OnS	1 S/M	-3.5(1.87)	-6.17(0.75)
	2 S/M	1.17(1.84)	-4.67(1.63)
	4 S/M	5.67(1.50)	-6.17(1.17)
BS	1 S/M	-3.5(1.05)	-2.33(1.50)
	2 S/M	86.67(2.34)	3.83(1.47)
	4 S/M	120.17(1.72)	13.67(1.97)
AS	1 S/M	-18.5(1.05)	-16.17(1.17)
	2 S/M	-6.67(1.37)	-15.67(1.21)
	4 S/M	5.67(1.21)	-15(1.41)

Table 5.5 shows the average and standard deviation of the meeting point error for each test case. The meeting point error represents the distance error between the stopping flag, which is 0.35 meters before the target, and the robot that does not reach the stopping flag first.

First, Table 5.5 shows how easy the AR can lose track of the TR when anticipation is not selected and the message frequency is low. For example, the meeting point error reaches 120.17 cm when the message frequency is 4 S/M and the TR is behind schedule. By comparing this value with the meeting point error under normal conditions, there is a difference of 123.67 cm which is a large distance considering that the test only lasts about 12 seconds. When the TS is moving ahead of schedule, the AR also shows a clear behavior of trying to speed up and get inline with the TS. In general, Table 5.5 shows that the anticipation module considerably reduces the meeting point error for all cases when the message frequency gets low.

Figures 5.8, 5.9 and 5.10 show sets of quartiles that describe the final position of the AR when the TR reaches the goal. The flag distance, that alerts the robots when

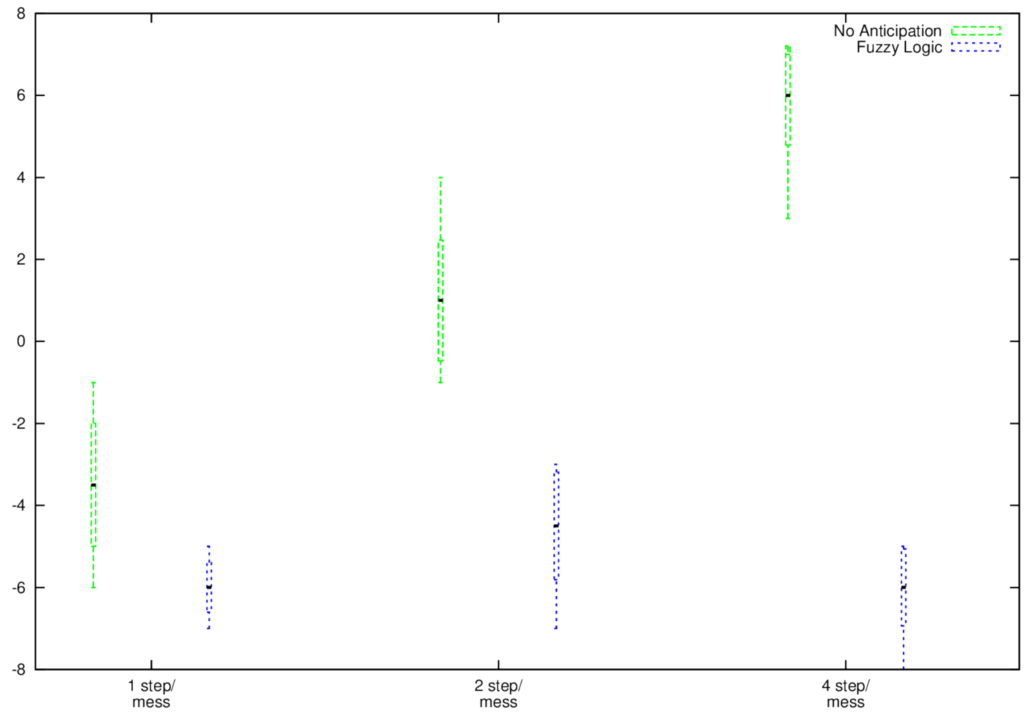


Figure 5.8: Average of the meeting point results when the TR is on schedule.

stop, was used as a reference point to know the last location of the AR at the end of the test. Both robots use the same value for the flag distance, so this approach helps to compare the data easily without having the robots driving in the same path and possibly colliding.

Figure 5.8 shows the behavior when the TR is on schedule. This figure gives an initial idea of the behavior of both robots when the TR is driving at its nominal speed. It can be seen that, even at nominal speed, the AR is not able to follow the TR while the message frequency gets lower. This can occur because the hardware of the robots is not completely identical and one robot can drive slightly faster that the other robot. But when anticipation is activated, the AR is able to get back on track and obtained a similar value for the meeting point from the test with 1 S/M.

Figure 5.9 shows the behavior when the TR is behind schedule. Here, the AR starts driving ahead of the TS but then, slows down and waits for it before reaching the goal. This figure (especially the third set of data) gives a clearer view of how easily

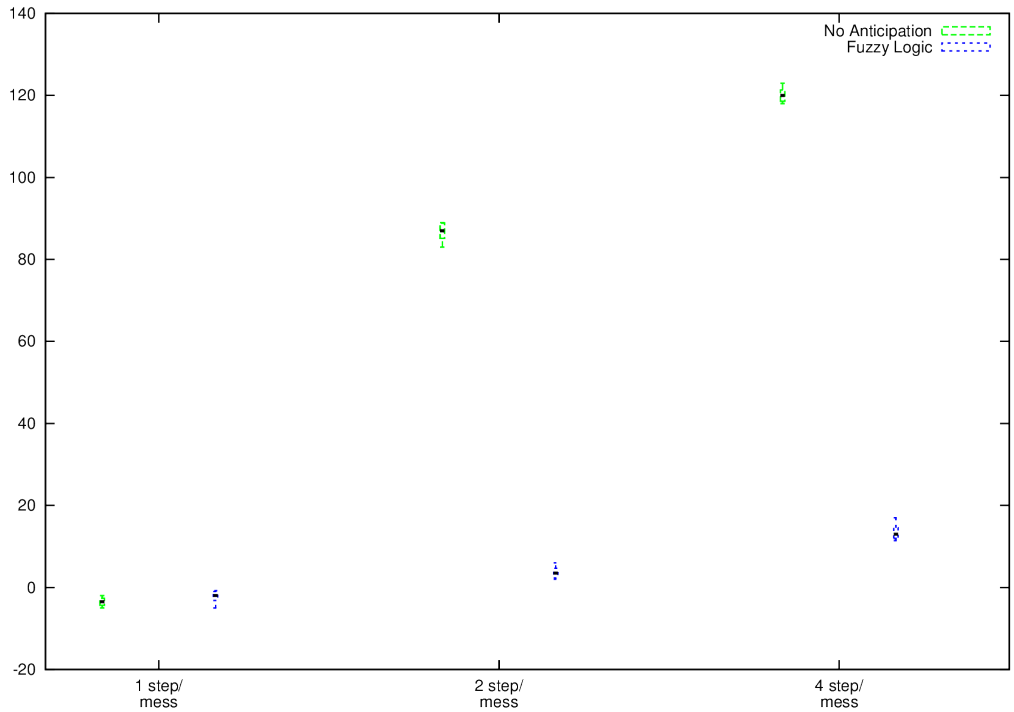


Figure 5.9: Average of the meeting point results when the TR is behind schedule. Here, the AR reaches the goal first, so the data represents the TR distance error.

the AR gets behind the TR when it does not get messages very often. Just by changing the message frequency from 1 S/M to 2 S/M, the average for the meeting point value has a variation of greater than 80 cm; and from 1 S/M to 4 S/M, the variation is even greater (about 120 cm). Figure 5.9 shows also that, with the anticipation module on, the AR is able to reduce this difference, significantly. Essentially, anticipation allows the AR to behave as if messages were being received at 1 S/M.

Figure 5.10 shows the behavior when the TR is ahead of schedule. The common behavior that was observed while using this TR behavior and 1S/M was that the TR would begin moving ahead of the AR, but the AR was able to speed up and catch the TR before reaching the goal often over compensating and reaching the goal before the TR (hence the negative values). This TR behavior does not create an error as large as the one created with the Behind Schedule behavior. Between 1 S/M to 2 S/M, the error gap is greater than 10 cm. The AR is able to correct its speed more

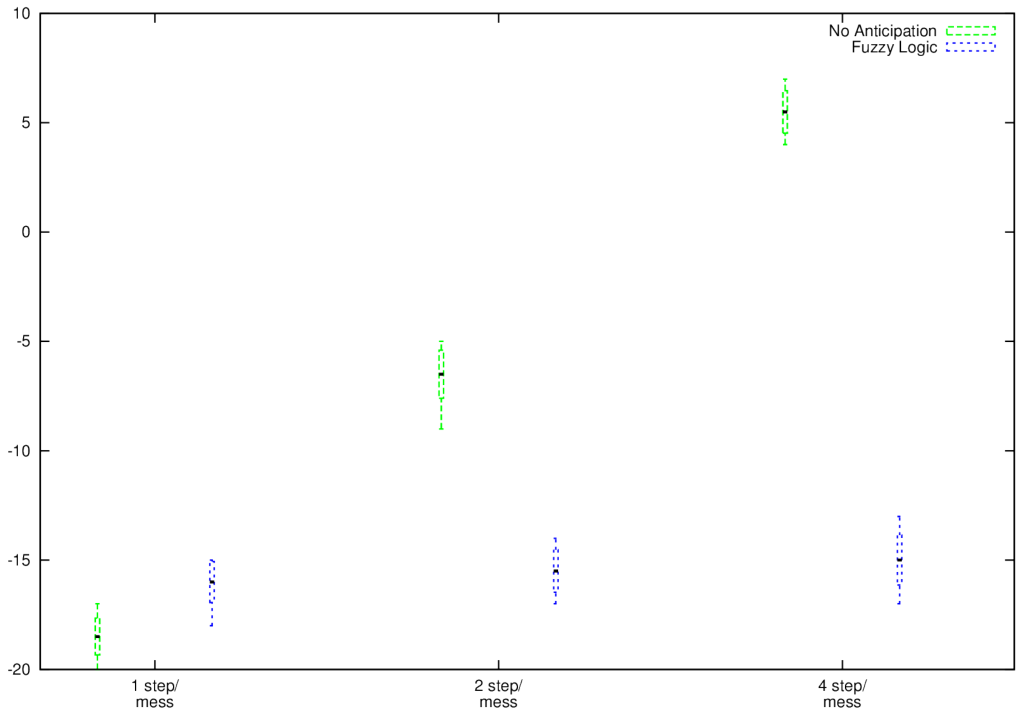


Figure 5.10: Average of the meeting point results when the TR is ahead of schedule. Here, the TR reaches the goal first, so the data represents the AR distance error

accurately while the anticipation module is on and reduce this error.

5.5 Conclusions

The data gathered from the robots when anticipation is deactivated shows that the AR robot looses track of the TR easily when the message frequency is low. Once the anticipation option is activated, the AR is able to reduce the error. In general, the behavior with anticipation is very similar to the behavior with No Anticipation and 1 S/M, showing that the anticipated messages are successfully filling in for the missing messages.

The experiments described in this chapter prove that this anticipation strategy is an effective method to infer messages and is able to recover the AR progress and synchronize it with the TR progress. Three different TS behaviors were tested with anticipation and, in all cases, the AR was able to meet the timing of the TR.

It can be concluded from these results that anticipation applied to robots in a real-world environment can effectively support the communication between two points when the communication link becomes erratic. This chapter showed an AI-based approach to anticipation that does not require a mathematical model of the environment to run the system effectively. This approach was able to overcome real-world problems that are commonly found while working machines, like added noise from sensors, which in this case, it was able to address without using additional filters to suppress them.

Chapter 6: General Conclusions

This research shows that anticipation strategies can be a very powerful tool when multiple agents try to communicate between each other. When an agent depends on the information generated by his peers to complete his task, missing information may force him to stop executing the task or cause errors or failures. Anticipation strategies allow the agents to continue executing their tasks effectively even while an agent cannot get in contact with his peers. Once the communication is reestablished, the agent's task will not be affected drastically and it will be easier to find the correct solution.

In this research, both a simulated and a real environment were created. Both environments included an anticipation module as an additional tool to support a set of mobile agents solving a cooperative navigation problem. The goal of the anticipation module is not to have full control of the agent or to serve as a controller for the agent's speed. Instead, its main function is to support the system with an anticipated message when a message from the lead agent is not received. This method allows the agent to behave in a more optimal manner when messages are dropped.

For the simulated environment, different anticipation models based on Neural Network and Fuzzy Logic models were generated and tested in order to analyze and compare their performance. Four different Neural Network models and one Fuzzy Logic model were used as anticipation models. The Neural Network models differ in the input neurons/number of past messages to analyze and the number of neurons in the hidden layer. By varying these two variables the following Neural Networks were created and tested: 10 input neurons/2 messages with 5 hidden neurons, 10 input neurons/2 messages with 10 hidden neurons, 25 input neurons/5 messages with 5 hidden neurons and 25 input neurons/5 messages with 10 hidden neurons. Additionally, the anticipation module was tested with two different configurations: anticipation without feedback and anticipation with feedback. The difference between these two configurations is in how the history of messages is updated for the inputs. The no feedback configuration only uses the real messages received from the TS; the feedback configuration uses the anticipated messages to updated the history of messages and uses them as inputs as well. To test all these possible configurations for the anticipation module, several different behaviors for the TS were created in which the TS changed its speed in different ways making it more or less difficult for the AUVs to follow its progress. A final feature used in the simulation was an adjustable message frequency. This feature allows control of how often the messages are sent to the set of AUVs, which directly affects the ability of the AUVs to follow the TS.

For the real environment, two robots were used to implement a similar behavior as in the simulated environment. Here, the leader (TR) also uses different profiles to control its speed and different message frequencies to control the communication with the second robot (AR). The AR uses a fuzzy logic model as the anticipation model. For this problem, both robots attempt to reach the goal marked by a green ball at the same time. The AR adjusts its speed according to the messages received from the TR. The anticipation module is in charge of filling any gaps left by a missing message from the TR.

Based on the results, the anticipation module is a non-invasive tool that helps to reduce the error considerably when it detects that there are missing messages in the communication. It is able to recreate similar messages, which help the speed controller choose the correct action to get to the meeting point on time. For the simulation, all five anticipation models work fairly similarly, but, in general, the Neural Network with 10 inputs and 5 hidden neuron and the Fuzzy Logic model had slightly a better performance for the different TS behaviors. These two models were able to help the fleet of AUVs meet the TS closest to the reference meeting point. Additionally, the anticipation model without feedback was able to reduce the error better than the model that uses feedback. Although both configurations were able to reduce the meeting point error, anticipation without feedback showed more accuracy and stability by having, in general, a smaller average error and a smaller standard deviation.

On the other hand, the results with the Neural Networks show that a Neural Network is able to learn from the training data how to generalize particular behaviors and apply them effectively to anticipate events that might affect the performance of a task. In this case, with proper training, the Neural Network was able to identify when the TS was slowing down or speeding up and, in addition, it was able to infer the correct magnitude of speed changes to have the fleet of AUVs synchronized with the TS. Because the Fuzzy Logic model uses hand-coded rules, there is no training or learning phase like in the Neural Network model. However, this does not mean that the Fuzzy Logic model cannot be configured to have a learning phase. A method like evolving the fuzzy set values can be considered as an option for future work with the fuzzy logic model. At the end, both models were successful in supporting the AUVs to complete the mission.

The results described for the simulated and real environment show that anticipation can work to improve a communication problems. But like any other system, it has limitations as well. Once the gap between the messages gets large enough, it would be very difficult for the anticipation module to infer possibly small behavior changes that occur during this gap and this is likely to affect the result. Just as with humans, anticipation will not be as effective if the actual communications are sparse and unrelated.

The anticipation strategy that was proposed for this research is a promising tool to support systems with communication problems or that run under extreme conditions where the information is likely to be lost, such as in very noisy environment or low bandwidth communications channels. It is worth noting that the communication does not need to focus on electronic package transfer systems. Communication can also be established through visual or audio signals. These methods can also be affected by noise or get interrupted and anticipation is potentially a helpful tool for these forms of communications as well.

References

[1] G.S. Atsalakis and K.P. Valavanis. Forecasting stock market short-term trends using a neuro-fuzzy based methodology. *Expert Systems with Applications*, 36(7):10696–10707, 2009.

[2] N. Baba and M. Kozaki. An intelligent forecasting system of stock price using neural networks. In *Neural Networks, 1992. IJCNN., International Joint Conference on*, volume 1, pages 371–377. IEEE, 1992.

[3] Y Uny Cao, Alex S Fukunaga, Andrew B Kahng, and Frank Meng. Cooperative mobile robotics: Antecedents and directions. In *Intelligent Robots and Systems 95.'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on*, volume 1, pages 226–234. IEEE, 1995.

[4] Robin Deits, Stefanie Tellex, Pratiksha Thaker, Dimitar Simeonov, Thomas Kollar, and Nicholas Roy. Clarifying commands with information-theoretic human-robot dialog. *Journal of Human-Robot Interaction*, 2(2):58–79, 2013.

[5] Gregory Dudek, Michael Jenkin, Evangelos Milios, and David Wilkes. Experiments in sensing and communication for robot convoy navigation. In *Intelligent Robots and Systems 95.'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on*, volume 2, pages 268–273. IEEE, 1995.

[6] Arthur C Graesser, Murray Singer, and Tom Trabasso. Constructing inferences during narrative text comprehension. *Psychological review*, 101(3):371, 1994.

[7] Anthony J Greene, William L Gross, Catherine L Elsinger, and Stephen M Rao. An fmri analysis of the human hippocampus: inference, context, and task awareness. *Journal of cognitive neuroscience*, 18(7):1156–1173, 2006.

[8] N. Hallin, B. Johnson, H. Egbo, M. O'Rourke, T. Soule, and D. Edwards. Simulating human reasoning in mine-like object inspection assignments for a formation of unmanned underwater vehicles. In *Advanced Technologies for Enhanced Quality of Life, 2009. AT-EQUAL'09.*, pages 158–162. IEEE, 2009.

[9] N. Hallin, B. Johnson, M. O'Rourke, and D. Edwards. A fuzzy logic resource optimizer for a fleet of autonomous vehicles in low-bandwidth conditions. In *OCEANS 2009-EUROPE*, pages 1–8. IEEE, 2009.

[10] N.J. Hallin, H. Egbo, P. Ray, T. Soule, M. O'Rourke, and D. Edwards. Enabling autonomous underwater vehicles to reason hypothetically. In *OCEANS 2009, MTS/IEEE Biloxi-Marine Technology for Our Future: Global and Local Challenges*, pages 1–7. IEEE, 2009.

[11] N.J. Hallin, J. Horn, H. Taheri, M. O'Rourke, and D. Edwards. Message anticipation applied to collaborating unmanned underwater vehicles. In *OCEANS 2011*, pages 1–10. IEEE, 2011.

[12] N.J. Hallin, B. Johnson, H. Egbo, M. O'Rourke, and D. Edwards. Using language-centered intelligence to optimize mine-like object inspections for a fleet of autonomous underwater vehicles. *UUST'09*, 2009.

[13] B. Johnson, N. Hallin, H. Leidenfrost, M. O'Rourke, and D. Edwards. Collaborative mapping with autonomous underwater vehicles in low-bandwidth conditions. In *OCEANS 2009-EUROPE*, pages 1–7. IEEE, 2009.

[14] M.T. Jones. *Artificial Intelligence: A Systems Approach.* Jones & Bartlett Learning, 2008.

[15] Y. Kara, M. Acar Boyacioglu, and Ö.K. Baykan. Predicting direction of stock price index movement using artificial neural networks and support vector ma-

chines: The sample of the istanbul stock exchange. *Expert systems with Applications*, 38(5):5311–5319, 2011.

[16] Hema S Koppula and Ashutosh Saxena. Anticipating human activities using object affordances for reactive robotic response. RSS, 2013.

[17] J.E. Laird. It knows what you're going to do: adding anticipation to a quakebot. In *Proceedings of the fifth international conference on Autonomous agents*, pages 385–392. ACM, 2001.

[18] Andrey K Morozov, Thomas W Altshuler, Clayton P Jones, Lee E Freitag, Peter A Koski, and Sandipa Singh. Underwater acoustic technologies for long-range navigation and communications in the arctic. In *Proc. 4th International Conf. Underwater Acoustic Measurements: Technologies and Results*, pages 1119–1126, 2011.

[19] Paul M Muchinsky and Arthur L Dudycha. Human inference behavior in abstract and meaningful environments. *Organizational Behavior and Human Performance*, 13(3):377–391, 1975.

[20] A. Rajala and D. Edwards. Allocating auvs for mine map development in mcm. In *OCEANS 2006 - Asia Pacific*. IEEE, 2007.

[21] A. Rajala, M. O'Rourke, and D.B. Edwards. Auvish: An application-based language for cooperating auvs. In *OCEANS 2006*, pages 1–6. IEEE, 2006.

[22] S.J. Russell and P. Norvig. *Artificial Intelligence: a Modern Approach*. Prentice hall, 2010.

[23] Satoru Satake, Takayuki Kanda, Dylan F Glas, Michita Imai, Hiroshi Ishiguro, and Norihiro Hagita. A robot that approaches pedestrians. 2013.

[24] Milica Stojanovic and James Preisig. Underwater acoustic communication channels: Propagation models and statistical characterization. *Communications Magazine, IEEE*, 47(1):84–89, 2009.

[25] Y.F. Wang. Predicting stock price using fuzzy grey prediction system. *Expert Systems with Applications*, 22(1):33–38, 2002.