Reconfiguration Control in Adaptive Networks

by Karin Sigurd

Submitted to the Department of
Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degree
of

Doctor of Philosophy
in Electrical Engineering and Computer Science
at the

Massachusetts Institute of Technology
June 2003

Author...
Department of Electrical Engineering and
Computer Science, May 2003

Certified by...
Sanjoy K. Mitter
Professor of Electrical Engineering
Thesis Supervisor

Certified by...
Jonathan P. How
Associate Professor of Aeronautics and Astronautics
Thesis Supervisor

Accepted by..
Arthur C. Smith
Chairman, Department Committee on Graduate Students

This page intentionally blank.

Abstract

Reconfiguration Control in Adaptive Networks

by Karin Sigurd

Distributed control systems are emerging as more robust and flexible alternatives to traditional control systems in several mechatronic fields such as satellite control and robotics. Instead of relying on one large unit with a centralized control architecture, one thus uses a parallel structure composed of many simple controllers collectively capable of performing the same task as the large unit.

Reconfiguration control involves cooperation, coordination and mutual adaptation and is relevant in a number of engineering problems such as formation control, multiagent learning and role allocation. In addition to being a key issue for using the distributed control paradigm to its full potential, reconfiguration control also offers a well delimited framework for addressing a number of interesting theoretical questions in distributed control such as those related to the overlapping notions of cooperation and coordination.

We propose a unified game theoretic approach to the problem of reconfiguration control which interprets node positions as strategies, identifies each configuration with the unique equilibrium of a game and sees reconfigurations as switches of games. Our approach is implemented in two different settings, both related to trajectory planning, and illustrated with simulation results.

In the first setting, we propose replicator learning as a multiagent learning algorithm which is a generalization of the replicator dynamics and show convergence in any finite dimension l of the average strategy to any desired strategy as a function of the chosen game matrix. We show how this result can be linked to collective motion in a subspace of $\Re^{l-1}$ resulting in a successive visiting of a set of waypoints.

In the second setting, we propose a novel total field collision avoidance algorithm of magnetic nature which permits a set of vehicles to reconfigure successively without knowing each other's positions; strategic sensor positioning makes sure that the vehicles do not sense their own fields.

Contributions of our research are a multiagent learning algorithm, a unified game theoretic framework for addressing reconfiguration problems, the identification of reconfiguration control as a problem common to several different fields but previously addressed with field-specific methods, the proposal of a definition of robustness in this context and, for the two trajectory planning settings in which our algorithm was implemented, two algorithms for distributed coordination and collision avoidance, respectively.

Thesis Supervisors: Professor Sanjoy K. Mitter and Professor Jonathan P. How

Acknowledgements

I would like to thank my advisors, Professor Sanjoy Mitter and Professor Jonathan How, for the support, experience and knowledge they have given me.

I am very grateful to Professor Sanjoy Mitter for his guidance and encouragement during my whole time at M.I.T. His focus on rigor and stringency combined with his aesthetic appreciation of mathematics as an art and of mathematical intuition allow him to be open to new lines of thought and to seeing general patterns in radically different application contexts - this has been both instructive and inspiring. Professor Mitter's perseverance and his broad knowledge of the literature in his own field and in adjacent fields have been of great support.

I am also very grateful to Professor Jonathan How who has encouraged me in the correlation of theory and practice in a true engineering spirit and has spurred inventiveness in finding the right hardware and the right methodology. His emphasis on balance between theory and application and concerns with the physical limits of the hardware have been instructive as have his structural thinking and methodic approach to finding the right questions to ask and his openness to adapting to the hardware situation.

I would also like to express my gratitude to the members of my thesis committee, Professor Vincent Chan and Professor John Tsitsiklis, for their helpful comments and suggestions and to Professor George Verghese, who read my thesis proposal at an early stage and provided insightful comments and encouragement.

The friendly professionalism of Rachel Cohen, Professor Mitter's administrative assistant, has made all contacts with her a pleasure.

Doris Inslee, LIDS' staff administrator, has always been the right person to turn to for extremely rapid and precise action in any administrative or practical matter - each time she finds the solution in no time.

Marilyn Pierce in the EECS Graduate Office is the point of reference for all graduate students - her expertise, humor and good judgment have been of the highest value.

I would also like to thank my fellow graduate students and office mates in 35-307: Alex Wang, Alvin Fu, Angelia Nedić, Asuman Ozdaglar, Chunmei Liu, Haixia Lin, Jun Sun, Li-Wei Chen and Tengo Saengudomlert, for creating such a friendly and nice working atmosphere in our office.

My parents, Ulla and Rolf Sigurd, have, as always and from the very beginning, been of the greatest support and I thank them very much.

This work was supported in part by the NSF-KDI Grant ECS-9873451, the Center for Imaging Science Subcontract Grant DAAD 19-99-1-0012 (with Johns Hopkins University) and the Department of Defense MURI Grant: Complex Adaptive Networks for Cooperative Control, Subaward # 03/132 (with the University of Illinois). The funding was very much appreciated.

Contents

Chapter 1

Introduction

Distributed control systems are emerging as robuster and more flexible alternatives to traditional control systems in several mechatronic fields such as satellite control [46] and robotics [3]. Instead of relying on one large unit with a centralized control architecture, one thus uses a parallel structure composed of many simple controllers collectively capable of performing the same task as the large unit. Robustness and adaptivity are some of the properties that make parallel systems superior to traditional ones; however, the distributed paradigm also poses a set of new challenges that need to be resolved in order to make the distributed paradigm fully operable. One of these is reconfiguration control, which is the topic of this thesis.

1.1 Reconfigurable Systems

Reconfigurable systems are part of the broader class of distributed systems; before introducing reconfiguration control, we will therefore comment briefly on the distributed paradigm and its benefits compared to traditional control.

The Distributed Paradigm Distributed control systems are as a rule robuster to local controller failure because of their parallel structure - other controllers can compensate for the lost ones - and because each controller is simple and inexpensive to replace. Furthermore, a distributed system has a higher surface to volume ratio, which facilitates interaction, sensing and detection. The spatial distribution of controllers also offers increased flexibility by allowing reconfigurations such as change of radii of a circular network functioning as the aperture of a telescope. However, to obtain these benefits it is necessary to address the question of coordination among the distributed units. How do the distributed units adapt to each other to achieve a collective goal although no unit has access to the whole system state? If one unit has a system failure, how do the other units detect this, know how to compensate for the failure and decide which units should do the compensation? Indeed, the study of coordination can also be motivated from a purely theoretical point of view since the overlapping notions of coordination and cooperation, currently attracting a great deal of interest, are yet to be given generally agreed upon definitions.

Cross-Disciplinary Field Fundamental issues related to the distributed control paradigm have for some time been simultaneously studied in several different fields, sometimes under different names. In control theory, large-scale power systems and air traffic control were the first main topics in distributed control, originally referred to as decentralized control as opposed to traditional, centralized control theory. In artificial intelligence, distributed systems are known as multiagent systems [14, 22] and appear mainly either as robotic systems or as softbot systems, that is, as virtual agents in computer simulations. A third type of distributed system related both to artificial intelligence and to biology is the swarm [4, 7], a system composed of a large number of simple and identical units. Natural distributed systems, such as natural sensor networks and immune systems, are to an increasing extent studied to serve as inspiration in engineering [17, 18]. Among the relevant notions in the different fields cited above are cooperation, coordination, emergence, self-organization, adaptivity and mixed initiative, further described in chapter 2. These notions, several of which are yet to be strictly defined, reflect a desire to make the distributed units adapt to each other to optimize their individual and collective performances and to achieve a desired system structure although no unit has complete knowledge either of the system state or of the desired structure. While the system needs to be adaptive and flexible, it also has to be robust against noise and failures in at least a fraction of the units.

Reconfiguration Control We will use the term reconfiguration control to denote the study of configuration keeping and change of configurations in distributed systems, as described in chapter three. By configuration, we understand any distinct structure assumed by the system and relevant for a given application and, by reconfiguration, a switch between any pair of such structures. While problems from this field have previously been addressed in very different contexts, such as formation control [12, 35, 50], multiagent learning and role allocation problems, these studies have as a rule addressed parts of the field and have offered solutions targeted at specific rather than general contexts. We propose a unified approach to the problem of reconfiguration control based on game theory which identifies each configuration with the unique equilibrium of a game and sees reconfigurations as switches of games. Our approach is implemented in two different settings, both related to trajectory planning.

Why Reconfigurable? It has been argued that a new control paradigm may be needed in order for distributed control systems to work optimally and make use of their full potential [43]. The fact that an optimal role allocation among the units can make a standard system operate as well as a more advanced system not operating at its optimal role allocation suggests that the systematic study of the control of reconfigurations may be of both theoretical and practical use.

1.2 Problem Statement

Having found the issue of reconfiguration to be a general problem in distributed control hitherto addressed only partially and with field-specific methods, we propose to formally define reconfiguration control, to formulate a general framework and algorithm applicable to any reconfiguration problem in distributed control and to demonstrate our proposed approach in a relevant application framework.

1.3 Contributions

The contributions of our work are as follows:

Unified Game Theoretic Framework Identifying reconfiguration control as an issue common to a range of different fields but previously addressed incompletely and with field-specific methods rather than with general ones, we proposed a general game theoretic framework for reconfiguration control that identified node positions with strategies, each configuration with the unique equilibrium of a particular game and saw reconfigurations as switches between such games, as described in chapter 4. An important part of this approach was to identify game classes that can be shown to have one and only one equilibrium, as was seen in section 4.3.

Replicator Learning We introduced Replicator learning as a generalization of the replicator dynamics [45] by making it adaptive in two layers rather than adaptive at the collective level and static at the individual level, as described in sections 5.2 and 5.3. Stability was proven in the case $l = 2$ for all matrices and in the general case $l > 2$ for a particular choice of the game matrix G, as seen in section 5.3.5. Furthermore, we derived a high-level trajectory planning algorithm, described in section 5.4, from this generalization that allowed distributed clustering around any of an uncountable set of points without any node knowing the position of any of the other nodes - this was seen as relevant in particular for a search application. By redefining the payoff functions for hetereogeneous nodes, the framework could also accommodate heterogeneous nodes, as described in section 5.5.

Total Field Collision Avoidance We proposed a novel approach to collision avoidance in multi-vehicle navigation in the form of a total field algorithm of magnetic nature, as seen in chapter 6; our algorithm allowed each vehicle to avoid collisions with the other vehicles without knowing their positions. By strategic positioning of magnetic field sensors orthogonally to the static field generated by the vehicle itself, each vehicle was able to measure the total field generated by all the other vehicles, as described in section

6.5. Furthermore, by choosing the sensor positions in a symmetric way, the magnetic field component contributed by Earth could be cancelled out in the calculation of field differences, as seen in sections 6.5.7 and 6.5.8. In section 6.6, it was shown how each vehicle could generate an estimate of the total field generated by the other vehicles based on the input from its sensors.

Robustness Definition We proposed in chapter 7 a definition of robustness in reconfiguration control as a system performance indifference to a large number of small node errors and to a small number of large node errors; this definition, given in section 7.2.3, was inspired by an analogy in estimation theory as described in section 7.2.2.

1.4 Disposition

In Brief Below, we first give a brief overview of the field of distributed control and point at reconfiguration control as an important part of this field. We then formally define reconfiguration control and describe the key questions to be addressed. After an introduction to game theory and its equilibrium notion, we present our proposed approach in general and illustrate its use in two different scenarios both related to trajectory planning. Finally, we address the notion of robustness in reconfiguration control, propose a definition of robustness and point at relevant current and future applications. A concluding section provides a summary and gives directions and suggestions for future research.

Distributed Systems In the chapter Distributed Systems, we introduce the emerging field of cooperative control in section 2.2 and discuss notions such as cooperation and coordination, teams and collectives, limited information, mixed initiative, adaptivity and differentiation, concluding in section 2.3 that reconfigurability makes a system adaptive and enables it to use its resources in an efficient way.

Reconfiguration Control Having identified reconfigurability as a desirable system property, we next define the notions of configuration and reconfiguration of a distributed system, motivate our definitions and give an example in section 3.1. Reconfiguration control then emerges as the union of configuration keeping and the controlled switching between configurations. In section 3.2 we propose a classification of reconfigurations into three major groups illustrated with examples and in section 3.3 we introduce the notion of configuration space, some relevant properties of which are identified. Finally, we point in section 3.5 at formation keeping, role allocation, multiagent learning and swarm control as the major fields of previous work in reconfiguration control and see that as a rule, previous approaches have offered solutions targeted at specific rather than general contexts.

A Game Theoretic Approach In the following chapter, we propose a unified game theoretic approach to reconfiguration control that interprets the positions of the distributed units as strategies, identifies each configuration with the unique equilibrium of a parametrized game and sees reconfigurations as switches of games. An introduction to game theory and its equilibrium notion is given in section 4.2 along with a motivation for a game theoretic approach in section 4.4 - its scalability, adaptivity, generality and robustness are found to be the main arguments in favor of a game theoretic approach. The two trajectory planning scenarios in which we propose to implement our approach are briefly introduced in section 4.3.4 as Replicator learning and Cluster, respectively.

Replicator Learning The Replicator learning scenario is a swarm scenario where a large number N of identical or similar simple units collectively achieve a result which is beyond the horizon of each individual unit. In the particular type of population dynamics known as the replicator dynamics, described in section 5.2, the system is adaptive only at the collective level, not at the individual level. We propose replicator learning in section 5.3 as a two-layered generalized replicator dynamics where both layers are adaptive.

Cluster The second scenario, Cluster, is a formation control scenario where a cluster of vehicles navigate safely in possibly changing formations visiting waypoints. We construct in section 6.4 a parametrized class of games that has a unique Nash equilibrium and introduce in section 6.5 a novel total field collision avoidance approach of magnetic nature which permits safe navigation without knowing the coordinates of any of the other vehicles. Strategic sensor positioning, presented in section 6.5.7, makes sure each vehicle senses only the field generated by the other vehicles, not any component generated by the vehicle itself, and also ensures that the magnetic field of Earth is cancelled out in the field difference calculated.

Robustness Having presented our general approach and its application in two specific scenarios, we next look at robustness in reconfiguration control. To formulate a definition of robustness in reconfiguration control, we first study the notion of robustness in control in general and how the current general definition evolved in parallel with the field of control. A definition of the notion of robustness in reconfiguration control is then proposed in section 7.2.3, inspired by an analogy in estimation theory as seen in section 7.2.2. Finally, the robustness of our approach applied to the two scenarios of Replicator learning and Cluster is studied in sections 7.3 and 7.4 and illustrated with simulation results.
Finally, a concluding section points at applications, gives a summary of the thesis and provides directions and suggestions for future research.

Chapter 2

Distributed Systems of Control

In this section, we give a brief introduction to some relevant notions in the field of distributed control and motivate the study of reconfiguration control as an important part of this field.

2.1 Introduction

A distributed control system is composed of $N > 1$ decision makers or controllers which typically do not share all information between them, are distributed in space and are interdependent as reflected by the system topology. Thus, a distributed control system is a network of controllers.

The field of distributed control has emerged over the last fifty years and is defined by the absence of central control, whereas classic control theory usually assumes a central controller. The need for decentralized or distributed control theory is to a large extent a result of the technological development which has shifted the attention from one single mainframe or power station to large networks of units where parallel processing is essential. Control theory, artificial intelligence, economics, network theory, cybernetics, biological systems and granular systems in mechanics are all fields where the distributed paradigm is important; game theory is an important theoretical discipline for analyzing distributed systems.

Learning While many classical equations describing population dynamics were formulated in the nineteen thirties - the predator-prey equations proposed by Volterra are one important example - the replicator dynamics was introduced in the nineteen seventies by Taylor and Jonker [45]. The analogy between the two-population version of the multi-population replicator dynamics and a two-player game was soon pointed at [52] and analogies were found between existing reinforcement learning algorithms and this version of the multi-population replicator dynamics [8].

Machine learning for single agents is one of the core subjects of artificial intelligence and adaptive control [5]. During the past decade, one has sought to investigate the more general decision problem of machine learning in multiagent systems where several independent agents try to make optimal decisions - multiagent systems are found in a rapidly increasing number of interesting application areas. One approach to machine learning in multiagent systems is to use single agent learning methods and treat the other agents as part of the environment, a method that may be too simplistic, however. A diametrically opposite approach is to model one's opponents in as great detail as one maps the effects of one's own behavior, a method which may lead to highly computationally complex systems. For these reasons, an intermediate way may be the best [32].

Game Theory Modern game theory was born in the nineteen forties with the publication of the book The Theory of Games and Economic Behavior by von Neumann and Morgenstern, with Borel, Cournot and Zermelo as important precursors in the late nineteenth and early twentieth centuries; Princeton was a center for game theory particularly during World War II. Nash, Shapley, Kuhn and Tucker made important contributions from the nineteen fifties and on; furthermore, Aumann, Harsanyi, Selten and Shubik have been important players from the nineteen sixties and on. In the sixties, approaches were proposed also for games of incomplete information.

Early on, game theory was concentrating on conflict solution in military and economic applications. However, after World War II game theory was studied rather by mathematicians than by economists [40]. In the nineteen seventies, game theory was discovered as a tool by biologists - at the same time, there was a revival of interest in game theory in economy. Today, game theory is used as a theoretical framework for behavioral science and learning. Could game theory also be of use for studying and describing machine learning in multiagent systems [32] ?

Small-World Networks Abstract network theory has many links to graph theory but was formally established with the work of Erdös and Renyi in 1960. In their random network theory, connections between nodes were assumed to be made at random, resulting in a network where most nodes have the same number of links to other nodes. Forty years later, Barabasi introduced the scale-free network as a network where a few hubs has a very large number of links to other nodes whereas most nodes have few connections; furthermore, Barabasi showed that a range of relevant networks such as the internet and social networks indeed belong in this category [2, 51].

Biological Networks The interest in artificial distributed systems is accompanied by a corresponding growing interest in biological networks, initiated by the foundation of the field of cybernetics in 1947 [53]. The nodes of biological networks consist of molecules such as proteins, of cells or of entire organisms and correspond to physiological networks, cellular networks such as the immune system or social networks such as schools of fish or flocks of birds. The systemic approach in biology is still in its beginning and, since the networks to be analyzed and modelled are generally very complex, efforts are often aimed at formulating concepts to begin to even qualitatively understand the dynamics and other properties of biological networks. Some fundamental concepts are the allowable size of the network, its connectivity and how the nodes actually interact - by studying simpler networks in detail, progress can be made in the formulation of relevant network notions.

2.2 Cooperative Control

The emerging notion of cooperative control refers to distributed engineering systems into which cooperation is hardwired. To achieve cooperation by design, it is essential to have clearly defined what cooperation is - in this section, we introduce the overlapping notions of cooperation and coordination and point at the notion of mixed initiative and the property of adaptivity as essential in cooperative control.

2.2.1 System Topology

The first issue is the system topology, indicating how the different parts of the system are connected and how they interact. Can all nodes communicate with one another, are there one-way communication links or possibilities to broadcast messages? Should distributed systems be designed in hierarchy with centralized subsystems or rather should even the simplest constitutive parts have a parallel, interconnected structure into which redundancy is hardwired?

Clone, Team or Hierarchy? Clones, collectives, colonies and teams are some notions encountered in cooperative control. The notion of a clone or a colony is close to that of the swarm, a collection of a large number of similar or identical units that collectively achieve results which are beyond the horizon of each individual unit. A team is a collective where the individual performance is only counted indirectly as part of the collective result. Opposite to all these parallel structures is the hierarchy, where upper layers control lower layers while receiving feedback.

Leaders and Followers Leaders and followers are parts of a hierarchical topology often used in formation control, where some leader nodes lead the way and serve as reference points for follower nodes, who may have less information about the system trajectory.

Static or Changing? Is the system topology static or changing? Although a leader-follower architecture is hierarchical, it assumes a parallel character if the nodes take turns as leaders, which may be the case if the leader position is an energy-consuming one.

2.2.2 Cooperation and Coordination

The notion of cooperation is often preferred to that of coordination in describing distributed system qualities, yet the terminology is not yet fully defined and the notion of cooperation is used to denote phenomena ranging from the collective achievement of a concrete goal to mutual adaptation.

Cooperation vs. Coordination Although the notions of cooperation and coordination overlap, the term cooperation is as a rule used to denote win-win situations where two or more units act so that they all perform better than each would on its own, whereas the notion of coordination denotes a mutual adaptation so as not to cancel out each other's actions or be in each other's way. This interpretation of cooperation also applies to situations where one unit would achieve nothing on its own because the task is too overwhelming, such as retrieving a heavy object.

Prisoner's Dilemma While the actual design of cooperative systems is still an emerging field [44], the theoretical notion of cooperation has been studied for decades, notably in the form of the classic game theoretic problem known as the Prisoner's Dilemma [1]. In this scenario, two prisoners and presumed accomplices are each faced with the choice of denying or confessing their guilt. Although their respective payoffs - punishments, in this case - are functions both of their own choice of action as well as of that of the other prisoner, they both ignore each other's choices.

$$\mathbf{G_{PD}} = \begin{array}{ccc} Player1/Player2 & Deny & Confess \\ Deny & (-1,-1) & (0,-10) \\ Confess & (-10,0) & (-8,-8) \end{array}$$

The payoff matrix $\mathbf{G_{PD}}$ is constructed so that the only game theoretic equilibrium, corresponding to both prisoners confessing, is Pareto minimal, whereas the strategy set that maximizes the average player performance is not an equilibrium. A large number of different learning algorithms have been tested where players try to achieve cooperation by using the information available in the observed previous actions of the opponent - the best known of these algorithms are Tit-for-Tat and Pavlov, which both are pure strategies conditional on the players' actions in the latest iteration of the game.

Proposed Definition We propose to define a cooperative system as a distributed system where, given that a performance measure is implied, the dynamics is such that for each node, the performance is higher than it would have been if each node had performed isolated on its own.

2.2.3 Limited Information

Characteristic of a distributed system is the limitation of information - no controller has a global overview.

Communication and Protocols Is the exchange of information essential for cooperation? Some feedback to the individual units seems necessary - the classic studies of the Prisoner's Dilemma assumed that the players could observe each other's previous actions and use them to try to predict their opponent's next action. However, the players were not assumed to make any mutual agreements about future actions. Is the exchange of information essential for coordination? Rather, it seems essential to have a set of network rules - a protocol - by which each node has to abide.

Stigmergy Stigmergy is a form of indirect communication through the environment encountered in natural systems in the form of scent trails. By making temporary or permanent changes to the environment, each node can send delayed signals to other nodes that will later come to the same location. In such a scenario, the envionment is thus dynamic [47] rather than static and is sometimes referred to as a smart environment.

2.2.4 Mixed Initiative

When no single controller has a global overview over the whole system, several controllers are likely to be in charge simultaneously.

What is Initiative? In the centralized control paradigm, both decision making and information are centralized and each system change is thus initiated by the centralized controller. In distributed control, system changes may be initiated by any of the nodes, a system quality denoted mixed initiative. To picture the notion of initiative and how it can switch between nodes, we may use a ball game such as soccer as an analogy. At any given moment, one player has the ball - from a dynamic point of view, that player has more influence than the others on the current development of the game and can be seen as having the dynamic initiative.

Parallel Initiative Continuing this analogy, should there be just one ball or several? Should each player be able to simultaneously handle only one or several initiatives? Is it better to have established rules that foresee most possible situations and conflicts that can occur or it is sufficient to let each player do what is best from his perspective? It seems clear that the latter alternative is not a realistic option at least when one wants system performance at the margins of the system capability. Initiatives in parallel may be present either to increase performance or as a redundancy to ensure robustness. How much redundancy is necessary and how is it related to robustness?

2.2.5 Adaptivity

With no central controller, mutual adaptation is necessary to avoid conflict or system collapse.

Intelligent System, Smart Gadget Although there may be many ways of defining an intelligent system, adaptivity is a property likely to be part of most such definitions. The words intelligent and smart are both encountered in cooperative control but in slightly different contexts - the word intelligent usually refers to an entire system which adapts to a range of signals in a complex but purposeful way whereas the word smart as a rule describes a concrete object that efficiently and quite directly reacts to one or a few signals by adaptation.

Adaptivity and Learning Adaptivity and learning are two overlapping notions for directed and purposeful change performed on-line. The term learning, used in artificial intelligence, is wide and typically applies to robot or softbot scenarios whereas the notion of adaptivity, preferred in control theory, originally denoted simple mechanisms for on-line parameter changes. Adaptivity is an essential property of a distributed system, but to hardwire adaptivity into a network one faces several challenges. Even when there is only one learner, it is important to focus on the goal - sometimes, several goals may be pursued simultaneously, but at other times, two goals may be mutually exclusive so that a split focus may lead to a zero net result. When the system is made up of many learners, the number of potential such conflicts rises dramatically. Not only do the learners have to take changing system goals into account, they also have to adapt to possible structure changes and flaws in the network itself, since other learners may make mistakes or learners may be added or lost.

Evolution The notion of evolution implies a long-term adaptation at the system level rather than at both the system level and the individual level and may typically involve several generations of nodes and a selective pressure.

Emergence and Self-Organization Emergent properties of a dynamic system are properties that may not initially be present but arise with time as as a natural outcome of the system dynamics itself. Emergence was originally mostly studied with heuristic methods in artificial intelligence, but lately, the topic has attracted attention in control theory where the aim is to prove emergence mathematically [9]. The notion of self-organization is closely related to the issue of emergence to which it adds the idea of purpose - the properties emerging in a given situation should not be random ones, but the ones most useful in that particular situation.

2.2.6 Differentiation

Differentiation transforms a homogeneous system into a heterogeneous one, composed of nodes that are temporarily or permanently specialized to fill a particular set of functions. When the capacities of at least some nodes overlap, we have a reconfigurable system, since those nodes can switch roles with each other.

Efficiency If a system is composed of identical units, it is easy to get an overview from any perspective of the available network resources. While this may not be as easy in a differentiated system, if for a given task an optimal or sub-optimal role allocation can be found, the system may operate very efficiently. By being able to reconfigure easily, a less advanced but differentiated network may thus be able to simulate a homogeneous non-reconfigurable network where all nodes are equally advanced.

Polymorphism Reconfigurability allows the system to show polymorphism both at the global level and at the individual level, since not all functional roles may have to be filled at the same time and different role allocations at the individual level may give different system properties. A polymorphous individual unit can fill more than one functional role or assume more than one shape or color; likewise, a polymorphous system can change its appearance, size or some other essential property.

Modularity Reconfigurability may also permit the system to be modular at the global level as well as at the individual level - the system may temporarily split into subsystems or modules that are later rejoined to form the original system. Thus, as each individual node may be composed of detachable modules - picture a mechatronic unit made up of separable functional parts - the whole system can split and merge.

Reconfiguration The two main components of reconfiguration control are configuration keeping and switching from one configuration into another - both parts give rise to a number of questions. Given that the system is in a configuration, are the units collectively responsible for keeping the configuration or are some more responsible than others? Do their responsibilities overlap or are they separated?

The issue of switching deals with initiative, which was discussed in the previous section. Can anyone initialize a reconfiguration or only some units? How is the initialization signaled to the other units? How are simultaneous and conflicting initializations avoided? How does one prevent noise from causing an initialization? Reconfiguration control certainly raises many questions and is the topic of this thesis.

2.3 Reconfiguration Control

In this section, we motivate the study of reconfiguration control in distributed control and comment on the choice of the term configuration in this context.

2.3.1 Term from Molecular Chemistry

In molecular chemistry, the term configuration is used to distinguish molecules of the same substance that differ only in the relative positions of some atoms. The distinction between these forms is made because it corresponds to a functional difference - the geometrical diversity of the two types of molecule gives them different properties at the macroscopic level. A simple example is the cis- and trans-configurations of 1,2-dichlorine cyclopropane shown in figure (2.1). To the left, we see the cis-version where the chlorine atoms are situated on the same side of the plane formed by the carbon ring whereas in the trans-version to the right, the chlorine atoms are situated on opposite sides.

Indeed, the field of chemistry also provides examples of both reversible and irreversible reconfigurations as molecules react and form new substances. Furthermore, from physics we know the reversible reconfigurations of substances that appear in different phases as solids, liquids, gases or plasma as a function of temperature and pressure.

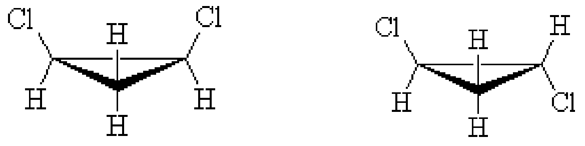

Figure 2.1: Cis-form of a 1,2-dichlorine cyclopropane molecule to the left, trans-form to the right.

2.3.2 Analogy in Distributed Control

Inspired by the use of the term configuration in the molecular chemistry setting cited above, we propose to use the same term to denote distinct sets of relative positions of nodes in a network. While the term formation refers to a spatial relative positioning of mechatronic units and has the connotation of order, the term configuration applies also to other contexts than spatial ones and fits networks of any size.

2.3.3 Motivation

What motivates our choice of reconfiguration control for a research topic? The potential for reconfiguration is a major key to the superior flexibility and robustness of distributed systems over centralized systems - the study of reconfiguration control can thus be motivated from an application-oriented point of view as a major issue to tackle in order to make the distributed control paradigm fully operable. The study of reconfiguration control is also motivated from a theoretical point of view since it is a well delimited subject in distributed control that yet incorporates many of the essential theoretical issues related to the distributed paradigm.

2.3.4 Reconfiguration Examples

To illustrate the practical use of reconfiguration control, we provide some examples below of reconfigurations in different applications.

Robotics Mobile robots of similar size and shape that can form reconfigurable clusters are of particular interest for manipulation and handling tasks [22] in the industry - one particular case is the handling of lamps for providing light. Robots for cleaning or clearing large areas are another application that may often involve operation in unsafe environments; if some robots are likely to be lost, reconfigurations that allow new robots to assume the places of the lost ones are essential.

Aerospace Small satellites forming rings [35, 46] of different radii can serve as distributed instruments such as telescopes. These instruments also have the benefit of being disposable since they can be reconfigured into dissolution as their final configuration after a completed mission. Another important application is the collective navigation in formations of fleets of small unmanned vehicles [26], which may form elastic mobile fences around ships or other travelling vehicles to protect them against intruders.

Computer Agents Role allocation [14, 22] among simulated computer agents forming a team as illustrated in robotic soccer is yet another relevant application area. As soon as there is some differentiation among players, different role allocations may make an important difference in the team performance. Since very dynamic domains may require frequent reconfigurations for optimality, efficient reconfiguration control may prove quite valuable.

MEMS Microelectromechanical systems (MEMS) for space applications would involve thousands of micro- or nanosatellites, passive or active, that could form a torus or a shell of varying radii around Earth and also be dissolved after a completed mission [36].

Chapter 3

Reconfiguration Control

Reconfiguration control is an emerging field in distributed control which deals with keeping a distributed system in one of a set of desired structures and making it switch between them in a controlled manner. As noted above, the choice of the term configuration was inspired by the analogous notion in molecular chemistry. Below, we propose definitions of the notions of configuration and reconfiguration, suggest classifications, comment on previous work and point at a set of central research questions in reconfiguration control.

3.1 Definitions

In the molecular chemistry setting referred to above, we saw that different configurations were distinguished by different relative positions of the constitutive parts. Wishing to extend the same principle to any distributed system, we formulate this idea mathematically in this section and propose definitions of the notions of configuration and reconfiguration.

3.1.1 Node Level

At any time t, each node k, $k = 1, ..., N$, is associated with a position $x^k(t)$ in a space X to which we refer as the node space - this position may indeed be the physical location of the node but may also correspond to something else, such as an activity level. The vector of ordered node positions $x^1(t)$, $x^2(t)$,... will be denoted $\mathbf{x}(t)$. The configuration is an emergent network quality and a result of the relative movements of the nodes in the node space.

3.1.2 Configuration

We propose the following definition of the notion of configuration:
Definition: a configuration C is a set of vectors $\mathbf{x} \in X^N$ that satisfy the configuration specific constraint $\mathbf{f_C}(\mathbf{x}) \in F_C$, where $\mathbf{f_C}$ may be a vector and

F_C is a set. Thus, $C = \{\mathbf{x} \in X^N \mid \mathbf{f_C}(\mathbf{x}) \in F_C\}$. The configurations are chosen disjoint; the set $C(t)$ formed by all possible configurations at time t is denoted the configuration space.
Any vector $\mathbf{x} \in X^N$ such that $\mathbf{f_C}(\mathbf{x}) \notin F_C$ for any C is said to be part of the **null configuration**.

3.1.3 Reconfiguration

We then use the proposed notion of configuration to define the concept of a reconfiguration:
Definition: a reconfiguration is a system switch in time $\Delta t < t_{max}$ from a position $\mathbf{x} \in C$ to a position $\mathbf{x}^{'} \in C^{'}$, where C and $C^{'}$ are distinct configurations and t_{max} is an application specific constant.

3.1.4 Motivation

We chose the above definition of a configuration to obtain a precise yet general expression that could allow for one global property being expressed in several alternative ways at the individual level and for possible indifference to rotations or permutations.

3.1.5 Example

We next give a simple example of a reconfiguring system of $N = 2$ nodes that can appear in two configurations in addition to the null configuration. Each node is associated with a node position $x^1(t)$ and $x^2(t)$ at time t; the nodes are considered to be in configuration $\mathbf{C_{Close}}$ if the distance $|x^1(t) - x^2(t)| \leq L_{min}$ and in configuration $\mathbf{C_{Distant}}$ if $|x^1(t) - x^2(t)| \geq L_{max}$.
Thus, in this case, $\mathbf{f_C}(\mathbf{x}) = |x^1 - x^2|$ is a scalar and

$$\mathbf{F_{C_{Close}}} = \{y|\ 0 \leq y \leq L_{min}\} \text{ whereas}$$
$$\mathbf{F_{C_{Distant}}} = \{y|\ y \geq L_{max}\}.$$

3.2 Reconfiguration Classification

In this section, we suggest a classification of reconfigurations into three different classes - permutation reconfigurations, system reconfigurations and structure reconfigurations - based on the network level at which the change takes place. Each class is briefly presented and illustrated with examples.

3.2.1 Permutation Reconfiguration

We will use the term permutation reconfiguration to denote the reconfiguration from a configuration $C \in C(t)$ to a configuration $C^{'} \in C(t + \Delta t)$ such that $C^{'}$ is a permutation of C. This is thus a reconfiguration only at the individual level, not at the system level, and corresponds to a change of roles between nodes. From a practical point of view, this reconfiguration class is important by letting the units take turns at occupying particularly stressful positions. Permutation reconfigurations may also be called for by an incomplete failure in an individual node, which is reallocated to a different position but remains part of the network. A mechatronic example would be a leader-follower navigation, where one unit at a time occupies the fuel-consuming lead position and the units replace each other at this position during navigation. A simple example of a permutation reconfiguration is shown in figure (3.1).

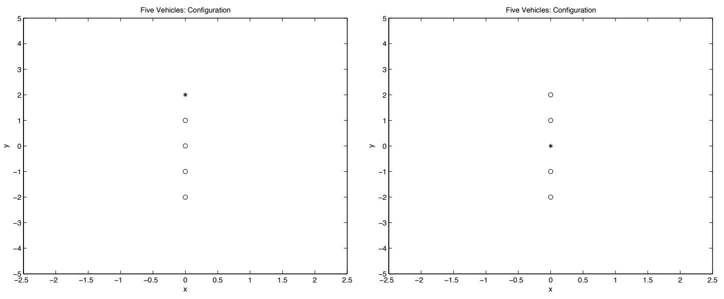

Figure 3.1: Example of a permutation reconfiguration where nodes 3 and 5 switch positions while all other nodes remain at their original positions.

3.2.2 System Reconfiguration

The term system reconfiguration will be used to denote the reconfiguration from a configuration $C \in C(t)$ to a configuration $C^{'} \in C(t + \Delta t)$ such that $C^{'}$ is not a permutation of C and $C \in C(t+\Delta t)$, $C^{'} \in C(t)$. Thus, a system reconfiguration is a reconfiguration both at the individual level and at the system level that occurs while the system hardware remains the same. This is the standard type of reconfiguration where the network goes through a reversible change to adapt by choosing a more appropriate configuration. For example, the small network below might change shapes from a V-shape into an I-shape in order to be able to pass through a narrow passage - such a system reconfiguration in shown in figure (3.2).

3.2.3 Structure Reconfiguration

A structure reconfiguration, finally, denotes a reconfiguration from a configuration $C \in C(t)$ to a configuration $C^{'} \in C(t+\Delta t)$ such that $C \notin C(t+\Delta t)$ or $C^{'} \notin C(t)$. This class of reconfigurations corresponds to a change of the very structure of the network, such as the addition or loss of nodes or the impairment of a node so that it can no longer fill any of the available roles. This type of reconfiguration requires more adaptivity of the network than the previous two configuration classes since in this case, we are dealing with

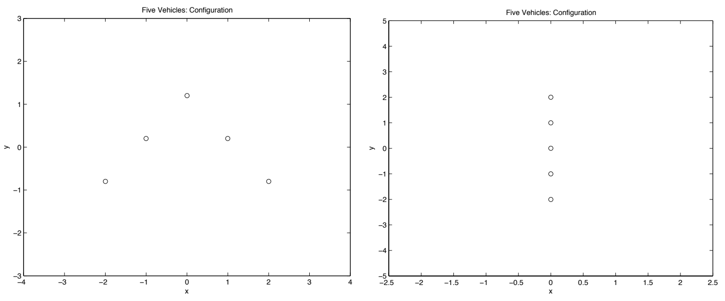

Figure 3.2: Example of a system reconfiguration from a V-shape to an I-shape.

more uncertainty - it may be difficult to predict at the design stage every possible structure change. This question is also closely linked to the notion of reconfiguration robustness, to be defined and discussed below, and to the question of irreversible reconfigurations and the change of the configuration space over time. Examples of structure reconfigurations are, in an economic network, the entrance of a new competitor in the market and, in a mechatronic setting, the addition of another squadron of vehicles sent out to join a fleet already on location. In figure (3.3), we see an example of a structure reconfiguration consisting in the addition of nodes to a network.

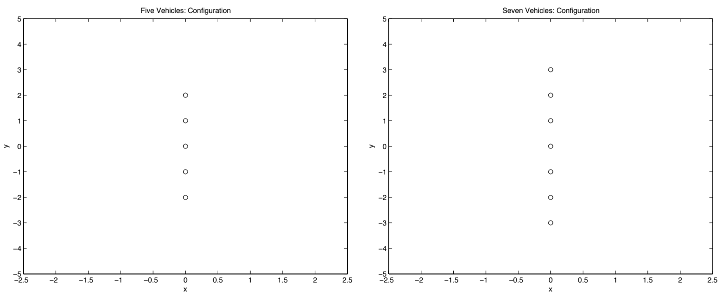

Figure 3.3: Example of a structure reconfiguration, where two nodes are added to an existing network.

3.3 Configuration Space

The configuration space at time t is the set $C(t)$ of possible configurations including the null configuration. In this section, we point at some criteria for classifying configuration spaces.

3.3.1 Node Space

The relative positions making up a configuration may be physical positions, points in time or have yet some other physical interpretation such as the individual degree of activity - this may influence some properties of the configuration space, such as its cardinality.

3.3.2 Static or Dynamic

Is the configuration space static over time or does it vary with time? If so, can one distinguish some structure such as periodicity in its variation? Many biological systems are tuned to the various rhythms associated with Earth such as the change of tides, light, temperature and magnetic field strength. Any system designed to operate in a periodically changing environment may need to compensate for these changes or adapt to them.

In a periodic configuration space, a known set of configurations recurs at a fixed rhythm. A qualitatively different situation is when a novel configuration is added to the set of configurations - it is a design problem to decide whether such additions will be necessary or not.

3.3.3 Conditional Configurations

Are some configuration options conditional upon the choice of previous configurations? Thinking of the set of configurations and possible reconfigurations as a set of nodes and the edges connecting them, respectively, this can be reformulated as the question of whether the set of configurations is a clique or not.

What are the advantages and disadvantages of having conditional configurations? Should particularly aggressive configurations be conditional ones, preceded by one and only one preparatory configuration to enhance synchronization and robustness?

3.3.4 Absorbing Configurations

From the application field of nanosatellite control comes the notion of disposable systems, that is, reconfigurable distributed systems whose last configuration is dispersal - a reconfiguration into an absorbing configuration is thus an irreversible reconfiguration. Are there other less obvious examples of absorbing configurations than dispersal? One such example might be the deadlock - a dynamic dead alley - where all nodes either freeze altogether or get caught in a loop.

3.4 Additional Issues

In the introductory section on distributed systems, we pointed at a number of new concepts associated with the distributed paradigm such as mixed initiative and limited information - next, we will see how these concepts apply to the particular field of reconfiguration control.

Reconfiguration Initiation

In reconfiguration control, mixed initiative occurs above all in the initiation of a reconfiguration from one configuration into another - the mechanism for reconfiguration initiation needs to be designed so that reconfigurations are not started unintentionally and so that intentional reconfigurations are yet performed swiftly.

While above we used the analogy of which player has the ball to illustrate the notion of initiative in a network, in the particular case of reconfiguration control we may aim at seeing a reconfiguration initiation as the pushing of a switch button accessible to all nodes.

3.4.1 Heterogeneity

As noted above, the presence of differentiation or heterogeneity among nodes makes efficient reconfiguration control particularly important since by successive reconfigurations, the network can adapt to changing exterior conditions and simulate a homogeneous network where the nodes are more advanced.

The heterogeneity on which we will focus concerns constraints on the positions that particular nodes can occupy in the node space - some nodes may perform optimally in particular intervals or not operate at all in others.

3.4.2 References

By references, we mean pieces of information available to some or all nodes about the network state such as the positions of key nodes - in formation control, the notion of leaders and virtual leaders is often used. One approach is to have only spatial distribution and let all information be shared between the nodes. However, we will address the situation where there is both spatial distribution and distribution of information - indeed, many researchers reserve the term distributed control for distributed systems of limited information and see networks of spatially distributed controllers that share all information as a version of centralized control.

3.4.3 Robustness

It is intuitively clear that the issue of robustness is most important in reconfiguration control - a configuration must not be dissolved because of noise or the failure of one or a few nodes. However, to be able to evaluate and compare the robustness of reconfiguring systems we first need to define the very notion of robustness in this context - in doing so, we will be inspired by the definition of robustness in other control settings.

3.5 Previous Work

Previous work in reconfiguration control has often been targeted at specific applications such as motion control of mechatronic clusters or computer agent interaction and offers solutions designed specifically for the chosen setting. The quite diverse fields of formation control, role allocation, multi-agent learning and swarm control are presented below as the major fields of previous work in reconfiguration control.

3.5.1 Formation Keeping

Formation keeping is a control application relevant in robotics and artificial intelligence, where groups of mobile robots are made to move in coordination so as to keep their relative positions constant; formation control is also becoming an important application field in aerospace control, especially for satellite control and unmanned aerial vehicle navigation [26].

Potential Field Formation keeping is often achieved by creating local potential fields for each unit based on the positions of all the other units, thus forcing each unit into its particular position in the formation - the formation is often symmetric so that each unit can use the same calculations [36]. However, each unit must have complete information of the positions of the other units and, furthermore, the initial positions of all units must not be too distant from the formation positions. The positioning into formation can be seen as a reconfiguration from an unstructured state into a well-defined configuration.

Leader-Follower Another major approach for formation keeping is the leader-follower approach [46, 50], briefly introduced above, where some units have more information than others and act as leaders. The other units aim to imitate the leaders or use them as reference points to which they should keep a fixed distance. This approach presents several centralized features and can be seen as a distributed control approach only if there are several leaders or if the units take turns at being leaders. As was the case above, this approach also requires the followers to have precise information about the position and possibly also the velocity of the leader.

Reconfiguration Reconfigurations between two ordered structures have been the focus of only a few studies [13], which have mainly addressed rotations - since the formation positions are often a local rather than a global equilibrium, an intermediate shift of positions may be needed before a different potential field may be applied. The inverse problem of concealing involontary cluster reconfigurations in a given reference frame has been addressed in order to make a system look invariant from Earth [12].

3.5.2 Multiagent Learning

Multiagent learning is a young discipline in artificial intelligence that seeks to generalize results from single-agent learning to the multiagent case [14, 32, 42]. Whereas convergence has been proven in the single-agent reinforcement learning case, analogous proofs are as a rule not available in the multiagent case.

Iterative Games The multiagent learning problem is often posed as the problem of attaining any of the equilibriums in an iterative game by learning from the payoffs obtained. The games may be simple zero-sum games, such as the game of matching pennies or rock-paper-scissors, or more challenging general-sum games. The question of Pareto optimality may also be addressed if there are more than one equilibrium.

The multiagent learning problem can be interpreted as a reconfiguration from an unstructured state to an equilibrium state. Since the issue is rather to reach any equilibrium than switching between equilibria, reconfigurations between structured states have hitherto been less often considered in this setting.

3.5.3 Role Allocation

The problem of allocating heterogeneous agents to static or dynamic functional roles in a multiagent system has long been studied in distributed artificial intelligence and tested in laboratory applications such as robotic soccer or collective box-pushing.

Fitness-Based Auctions Auction algorithms are often used, with or without a centralized broker or coordinator, to allocate a set of tasks to a multiagent system. The agents bid for tasks based on their particular functional profile and on whether they are currrently idle or busy [22] - if there is no broker, bids are broadcast and all agents are presumed honest. In this context, agents sometimes do switch tasks in synchronization, as when robotic soccer players switch roles in the game or take turns playing. However, in general, the switching is asynchronous, thus resulting in overlapping system reconfigurations of a different and somewhat more general nature. Negotation can take time but may, if implemented efficiently, be quite rapid. However, it may be more difficult to reduce the fault tolerance necessary for auction-based role allocation to work.

Allocation by Learning An alternative to auctions and negotation is learning, which was described above for general and sometimes quite simple configuration assumption. Heuristic multiagent learning has successfully been applied to role allocation problems such as elevator control [14].

3.5.4 Swarm Theory

Swarm theory was established in the late nineteen eighties by Beni [4] as a field in artifical intelligence, growing out of the work of von Neumann on cellular automata and inspired by complex natural systems made up of large numbers of seemingly simple cells or organisms such as bees or ants [7].

Flocking The first approaches in this field were heuristic and aimed at flocking, that is, keeping a static or moving distributed system coherent [41]. Each individual unit, referred to as a boid, used a small number of elementary rules to move based on the current positions and orientations of its closest neighbors, variables thus assumed to be available to each unit. The problem of keeping a swarm coherent can indeed be seen as a configuration keeping problem.

Proving Cohesion In the last few years, there has been a renewed interest in approaching swarm problems mathematically in order to prove that the application of a set of local rules will lead to the emergence of a desired system property such as cohesion. Passino assumed an attraction between nodes at long distances and a repulsion at short distances to prove swarm coherence and to show that the nodes would ultimately come to a stop within the swarm [20, 21].

3.5.5 Robustness

Robustness in distributed control deals with the effect of inaccuracies at the individual level on the system performance. Since reconfiguration control is still an emerging part of distributed control, existing studies deal more with designing algorithms for configuration keeping and reconfiguration than with investigating how the divergence from these algorithms affects the collective performance.

Node Failure As noted in the introduction, one of the major strengths of the distributed paradigm is the inherent robustness given by a parallel structure; this applies particularly to robustness against individual node failure. Role allocation problems often take into account the possibility of mistakes at the individual level - with an auction approach that uses time-limited contracts, an unfinished task can be put on the market again once detected. McInnes [35] considers the possibility of total node failure in his local potential function for formation keeping - the network will adjust to the loss of individual nodes. Little attention is given to incomplete node failure, where a node is still part of the network but with altered dynamics

- such a situation can be challenging since the node may cause prolonged system disturbance.

Communication Failure How can a tradeoff be made between the robustness offered by a protocol and the flexibility given by on-line communication? A conservative approach consists in always taking the worst-case scenario into account and leaving margins large enough even for the worst possible case. However, this approach goes against the distributed paradigm by using the system at only a fraction of its potential and by not taking advantage of the robustness to individual node failure - provided it is rare enough, the worst-case scenario does not have to be safe.

3.6 Research Questions

What are the main questions that we wish to address in reconfiguration control?

- **Generality** Can we find a general algorithm for reconfiguration control that would be valid in any of the diverse fields of application cited above?
- **Coordination** How can one assure that all units are coordinated although none of them knows the current position of any of the other units? What is the difference between coordination and cooperation?
- **Stability** How can we assure that a configuration is kept once it has been attained?
- **Initiative** What is the best system architecture for mixed initiative control? How can one prevent deadlocks and avoid too many initiatives at a time? Are some hierarchical features necessary or is it possible to use a purely parallel architecture?
- **Structure Change** How should changes in the network structure such as the addition or subtraction of nodes or partial failure in one node be handled?
- **Heterogeneity** Can node heterogeneity be incorporated in a framework originally designed for a homogeneous system with marginal changes?
- **Robustness** How should robustness be defined qualitatively in reconfiguration control? What are good quantitative measures of robustness in this context?

Chapter 4

A Game Theoretic Approach

In this section, we present a unified game theoretic approach to reconfiguration control. After a brief overview of our proposed approach, we give a short introduction to game theory and motivate why a game theoretic approach is appropriate. Our proposed approach is then described in detail and two particular scenarios are introduced in which the proposed algorithm is implemented.

4.1 Approach in Brief

We propose a game theoretic approach to reconfiguration control which interprets node positions as strategies, identifies each configuration with the unique equilibrium of a game and sees reconfigurations as switches from one game to another.

In this framework, permutation reconfigurations are seen as the exchange of payoff functions between individual players whereas system reconfigurations correspond to changes of game parameters for all players. Structure reconfigurations, finally, are interpreted as changes of the very format of the payoff functions, concerning such issues as the number of players N. The proposed framework also accommodates heteogeneous players by designing special payoff functions for such players.

4.2 Introduction to Game Theory

Below, we will first describe what defines a game and then introduce the central equilibrium notion in game theory, the Nash equilibrium, and one of its refinements, the ESS.

4.2.1 Defining the Game

Players, Strategies and Payoffs A game is defined by the number $N > 1$ of players taking part, by the strategies available to the players and by the payoffs given to each player for each possible combination of these strategies [37, 40].

Support Each player chooses his strategy p in the set $\mathcal{P}(A)$ of probability distributions over the action space A. The support of a strategy p is denoted $R(p)$ and defined as $\{i \in A \mid p_i > 0\}$, where $p(\cdot)$ is the probability of choosing a particular action.

Pure or Mixed? If $|R(p)| = 1$, p is known as a pure strategy, whereas if this is not the case, the strategy is denoted a mixed strategy. If $R(p) = A$, p is known as a totally mixed strategy.

4.2.2 Nash Equilibrium

Definition A Nash equilibrium of an N-player game of action space A is defined as a strategy vector $\mathbf{p} = (p^1, ..., p^N)$ where for each $k = 1, ..., N$, p^k is a probability distribution over A and the following condition holds for the expected payoff $E_k(\cdot; ...)$ to each player $k = 1, ..., N$, q being any other available strategy:

$$E_k(p^k; p^1, ...p^{k-1}, p^{k+1}, ..., p^N) \geq E_k(q; p^1, ...p^{k-1}, p^{k+1}, ..., p^N) \qquad (4.1)$$

Here, $E_k(p^k; p^1, ...p^{k-1}, p^{k+1}, ..., p^N)$ denotes the expected payoff to player k playing strategy p^k if the other players $i \neq k$ play p^i.

Strict or Weak? If the inequality (4.1) is strict for all k, the Nash equilibrium is a strict one whereas otherwise, it is denoted weak.

Number of? Each game for which $l = |A| < \infty$ has at least one Nash equilibrium [38].

4.2.3 ESS

Evolutionarily Stable Strategy ESS stands for Evolutionarily Stable Strategy and is a refinement of the Nash equilibrium designed as the central equilibrium notion of evolutionary game theory [33]; it is usually defined for two-player games.

Refinement of Nash Equilibrium The ESS refines the Nash equilibrium by requiring symmetry, that is, that both players play the same strategy at equilibrium. Furthermore, for a weak and symmetric Nash equilibrium to be an ESS, the payoff to a player who plays the equilibrium strategy against any other strategy has to be strictly larger than the payoff that the player would have received, had he played the alternative strategy against itself.

Definition An ESS of a two-player game of action space A is defined as a strategy vector $\mathbf{p} = (p^1, p^2)$ where $p^1 = p^2 = p$ is a probability distribution over A and either of the following conditions holds for the expected payoffs

$E(\cdot,\ \cdot)$, q being any other available strategy:

$$E(p,p) > E(q,p) \text{ or} \tag{4.2}$$

$$E(p,p) = E(q,p) \text{ and } E(p,q) > E(q,q). \tag{4.3}$$

Number of? A finite game can have no ESS, one ESS or several ESS's. However, if an ESS is totally mixed, it is the unique ESS of the game, as we will see below.

Game of Hawks and Doves As an illustration of the two equilibrium notions above, the matrix $\mathbf{G_{HD}}$ below defines a two-player, two-action game which has three Nash equilibria: $\mathbf{p}_1 =\{(1,0),(0,1)\}$, $\mathbf{p}_2 =\{(0,1),(1,0)\}$ and $\mathbf{p}_3 =\{(0.5,0.5),(0.5,0.5)\}$. Only the last one is also an ESS.

$$\mathbf{G_{HD}} = \begin{array}{ccc} Player1/Player2 & Action1 & Action2 \\ Action1 & (-1,-1) & (2,0) \\ Action2 & (0,2) & (1,1) \end{array}$$

This classic game is known as the game of Hawks and Doves with the two available pure strategies corresponding to aggressive and defensive behaviors, respectively.

4.2.4 Information

To illustrate how the set of equilibria of a game can be transformed by the information available to the players, we introduce information in the form of an asymmetry in the simple game of Hawks and Doves above.

Let us assume that at the beginning of each game, each player is assigned either of two possible roles $i \in I = \{1, 2\}$, that they both always have different roles in a given game and that the probability of either player being assigned a particular role in any game is 0.5. The strategies of each player will now be conditional on the information obtained at the beginning of the game. In fact, the set of Nash equilibria will now be enlarged to include two conditional, symmetric equilibria $\mathbf{p}_4 = \{p_4(\cdot|i), p_4(\cdot|i)\}$ and $\mathbf{p}_5 = \{p_5(\cdot|i), p_5(\cdot|i)\}$, where

$$p_4(j|i) = 1 \text{ if } i = j \text{ and } p_4(j|i) = 0 \text{ if } i \neq j \text{ whereas}$$
$$p_5(j|i) = 0 \text{ if } i = j \text{ and } p_5(j|i) = 1 \text{ if } i \neq j.$$

Indeed, both $\mathbf{p}_4$ and $\mathbf{p}_5$ are not only Nash equilibria but also ESS's. The addition of information to a game can only extend the set of equilibria, never reduce it, since it is always possible to disregard the added information. However, if it is assumed that no player chooses to ignore the additional information, the addition of information corresponds to a transformation rather than an extension of the set of equilibria.

The described extension of the game of Hawks and Doves also has a biological interpretation with the asymmetric roles being interpreted as those of the owner of a territory and of an invader, respectively. Strategy $\mathbf{p}_4$, consisting in being aggressive as the owner and defensive as the invader, is known as the Bourgeois strategy whereas its opposite, strategy $\mathbf{p}_5$, may be denoted the Anti-Bourgeois strategy.

4.3 Our Approach

We will now describe our proposed approach where each configuration corresponds to the unique equilibrium of a particular game and a reconfiguration is seen as a switch of games.

4.3.1 Configuration as Unique Equilibrium

We propose to see the node space of the individual nodes as a strategy space and to interpret each node position $x^k(t)$ as the strategy played by the node in a game. The set of configurations will correspond to a set of games of unique equilibria, where in each case the equilibrium strategies will be the desired relative node positions of the corresponding configuration. The system dynamics, finally, will be chosen to have this unique and parametrized equilibrium as the global attractor regardless of the parameter values.

4.3.2 Reconfiguration as Switch of Games

With configurations corresponding to the unique equilibria of different games, a reconfiguration will be interpreted as a switch of games and implemented as a switch of game parameters. While the game parameters will in the standard case correspond to the entries of a game matrix, we note that, as seen above, the addition of information to a game may be another valid way of transforming the set of equilibria of a game, if it is assumed that no player chooses to ignore the added information.

4.3.3 Games of Unique Equilibria

In many recent multiagent learning studies, a particular game is chosen and the convergence of the strategies to any member of the set of equilibria of that particular game is studied. Here, instead, a set of games is constructed that fits a particular equilibrium pattern and rather than switching between different equilibria of the same game, players switch between the unique equilibria of different games. For this approach to be efficient, it is necessary to identify classes of games and equilibrium notions such that each class has one and only one equilibrium of the chosen type and such that all the games in the class can be described in a compact and uniform way.

4.3.4 Two Scenarios

To illustrate our proposed approach, we will introduce two different scenarios in which our approach is implemented. The first scenario, referred to as Replicator learning, applies to swarms, where the aim is to make precise statments about the dynamics of the average position of a large number of simple and similar units. The second scenario, denoted Cluster, is a traditional formation control framework where a cluster of vehicles travels in formation to visit a set of waypoints while avoiding collisions. Before describing the scenarios in detail, we give a motivation for a game theoretic approach to reconfiguration control.

4.4 Motivating a Game Theoretic Approach

What motivates a game theoretic approach to reconfiguration control?

Scalability As a distributed approach that makes each node responsible for a large part of its trajectory planning in the node space, a game theoretic approach scales well. The number of computations required per node is as a rule constant, whereas the memory required per node may grow as $O(N^2)$, where N is the number of nodes. However, this memory requirement may be overcome by the addition of a blackboard table accessible to all nodes.

Adaptivity The fact that each node is responsible for a large part of its trajectory planning in the node space makes it easier to further decouple nodes without having to recompute the trajectories of all nodes, as might be necessary in a centralized approach. The game theoretic approach also adapts well to extended communication between nodes, since this merely adds equilibria to the game - the optimality of the added equilibria depends on the information exchanged. By assuming that no player chooses to ignore the added information and by seeing the added equilibria as equivalent, the same algorithm can still be applied.

Generality The game theoretic approach can be applied in any reconfiguration control setting and thus offers a unified theoretical framework for a range of different distributed control problems.

Robustness If robustness is measured as the worst-case configuration deviation caused by failure in one controller, then the centralized single-controller approach seems quite vulnerable since a controller failure may cause a complete system failure. Are all distributed algorithms then necessarily more robust? No, since some purely local distributed algorithms that assign the role of leaders to some nodes and the role of followers to the majority of nodes may suffer from a similar sensitivity: a follower failure is negligible whereas a leader failure may lead to a local squadron failure that can have at least temporary repercussions in the whole system.

However, the game theoretic approach assigns to each agent a local role in a global framework, thus in fact adding a centralized aspect to an otherwise distributed structure. Whereas in a purely local distributed framework, a local node failure will in the best case be equivalent to a local structure failure, the global aspect of the game theoretic framework makes it possible for other nodes to compensate for the local failure and accommodate it in a locally different but globally equivalent structure, thus cancelling out the failure.

Chapter 5

Replicator Learning

In this section, we describe the first of the two reconfiguration scenarios to which we propose to apply our game theoretic approach. We give a game theoretic interpretation of a trajectory planning problem and introduce replicator learning as a generalization of the replicator dynamics that can under certain conditions be shown to converge to the same equilibria as the single-population replicator dynamics; these equilibria are then matched to waypoints to visit. Finally, we show how this framework adapts to the presence of heterogeneous nodes and illustrate our results with simulations.

5.1 Problem Statement

Our system is composed of $N \gg 1$ units $k = 1, ..., N$, each associated at any time $t = 0, 1, 2, ...$ with a current position $x^k(t) \in \Re^m$, $m < \infty$, in a given m-ball B^m of arbitrary but finite radius R in the common coordinate system X. These positions are updated according to the same equation, where $f(\cdot)$ is some function and $G(t)$ is a set of parameters:

$$x^k(t+1) = x^k(t) + f(x^k(t), \hat{\bar{x}}_{-k}(t), G(t)), \text{ where}$$

$$\bar{x}_{-k}(t) = \tfrac{1}{N-1} \textstyle\sum_{i \neq k} x^i(t) \text{ and } \hat{\bar{x}}_{-k}(t) \text{ is an estimate of } \bar{x}_{-k}(t).$$

Given a set of waypoints y_1, $y_2, ..., y_M \in B^m$, we wish to make the average position $\bar{x}(t) = \frac{1}{N} \sum_{k=1}^{N} x^k(t)$ visit these waypoints in order although each unit is informed only of its own position in X and knows neither the positions of the other units nor the waypoints. Our task is thus to find a function $f(\cdot)$, an estimate $\hat{\bar{x}}_{-k}(t)$ and a parameter set $G(t)$ that make this possible.

5.1.1 Problem Motivation

Why is this an interesting problem? Firstly, it is a simple model for studying cooperation and coordination in a network of N units trying to achieve a collective goal, in this case positioning themselves relative to each other in such a way that the average position coincides with the current waypoint.

Secondly, the most direct application scenario is a search scenario, where a current estimate of the location of a lost object is available and corresponds to the current waypoint. In this scenario, not only does one want the average position of the searching units to coincide with the estimated location of the lost object; to maximize the observation area of the collective, one also wants to spread out the units around the waypoint.

We will comment more on the choice of the name replicator learning for this algorithm below; what is learned by each unit in this scenario is a position relative to the other units that makes the average position coincide with the desired waypoint. Thus, the whole network of N units collectively learns an average position.

5.1.2 Network Definition

The network is in this scenario composed by the N mobile units, who adapt to each other's positions to ensure that they collectively achieve an average position situated at the current waypoint.

5.1.3 Configuration Definition

In this scenario, the configurations are the set of possible waypoints, that it, the set of possible values assumed by the average position of the swarm, whereas a system reconfiguration is the switch from one such waypoint to another. We note that in this context, the configuration space is uncountable and we have many degrees of freedom for each possible configuration, that is, each configuration at the system level corresponds to a large number of possible combinations of node positions.

Using the configuration definition introduced above, $\mathbf{f_C}(\mathbf{x})$ is here a scalar and defined as

$$\mathbf{f_C}(\mathbf{x}) = \frac{1}{N}\sum_{k=1}^{N} x^k,$$

with $F_C = \{y_C\}$, or if some imperfection is tolerated,
$F_C = \{y \in X| \; |y - y_C| < \epsilon\}$ for some $\epsilon > 0$.

5.1.4 Proposed Approach

Our proposed approach is based on results from game theory and population dynamics. The replicator dynamics is a common type of population dynamics used to describe the relative propagation of different phenotypes, the relative fitness of which is given in the form of a game matrix representing a symmetric bimatrix game. By phenotype, we understand a distinct version of a given trait or physical property; the relative fitness of two phenotypes indicates how well the two phenotypes would do relative to each other in a competition for some relevant asset such as food, space or water.

Replicator Dynamics and ESS The individual population members are seen as discrete replicators that propagate a fixed phenotype, thus giving rise to a hybrid system which is adaptive at the collective, continuous level but static at the individual, discrete level. The ESS, introduced above, is known to be an attractor in the replicator dynamics. Under certain conditions, a game can be shown to have one and only one ESS, which will be the global attractor in the replicator dynamics.

Mapping Positions to Compositions We propose to map the positions of the N distributed units to the compositions of N disjoint populations growing according to the replicator dynamics. The dynamics of the average composition will then be seen to correspond to a modified replicator dynamics which under certain conditions converges to the same equilibrium as the replicator dynamics - this is shown analytically for any matrix for the case $l = 2$ and for a particular choice of G for $l > 2$. By further mapping each waypoint to the unique ESS of a different game and choosing the set of parameters $G(t)$ to be the current game matrix, the average position will converge to the desired waypoint, while a switch of waypoints will correspond to a switch of game matrices $G(t)$. Thus, we will have a bijective mapping between positions in a simplex in Euclidean space and compositions in a probability simplex.

By imposing an upper bound $v_{\max}$ on $|f(\cdot)|$, our problem can be seen as a motion control problem where $x^k(t)$ is the physical position of a mobile unit k moving in discrete time at speed

$$|x^k(t+1) - x^k(t)| = |f(x^k(t), \hat{\bar{x}}_{-k}(t), G(t))| < v_{\max}$$

for a system-specific constant $v_{\max}$. How, then, do we propose to give a game theoretic interpretation to the described trajectory planning problem?

Game Theoretic Interpretation of Trajectory Planning Problem The set of orthogonal axes of motion, parallel to the coordinate axes, will correspond to the game theoretic notion of an action space, which in population dynamics is represented by the set of distinct phenotypes. Thus, the physical position of each mobile unit will be interpreted as its current strategy in a game. The random interactions between units will consist in each unit k indicating a choice of a coordinate axis which indirectly reflects its current position. Each unit k then uses the opponent's choice to estimate the average position of the other units $\bar{x}_{-k}(t)$ and calculates $f(\cdot)$ as a function of this estimate $\hat{\bar{x}}_{-k}(t)$, of its own position $x^k(t)$ and of the cost parameters $G(t)$. Before describing our proposed approach in detail, we will introduce some notions from game theory and population dynamics that will be needed in the presentation of our approach.

5.2 Replicator Dynamics

The replicator equations [45] describe the relative propagation with time of a finite number of phenotypes in a population of replicators, given the assumption that each subpopulation representing a phenotype grows exponentially at a rate proportional to the fitness of the phenotype in the current population. While originally formulated as a biological model, the replicator dynamics also fits a game theoretic framework, as we will see below.

In this section, we give the replicator equations in continuous and discrete time, point at their biological and game theoretic interpretations and comment on their convergence properties.

5.2.1 Single-Population Replicator Dynamics

The standard version of the replicator dynamics, usually called just the replicator dynamics, is more correctly referred to as the single-population replicator dynamics, since in this model all replicators belong to the same population. We first present this standard form of the replicator dynamics and then a generalization in the form of the multi-population replicator dynamics.

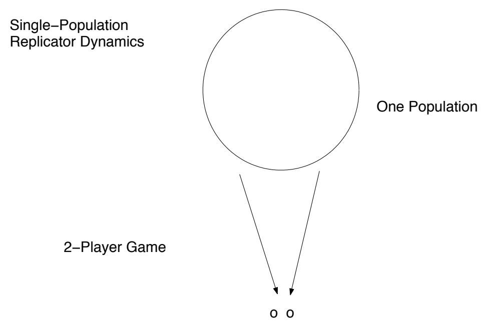

Figure 5.1: The single-population replicator dynamics.

Continuous Replicator Dynamics

The single-population replicator dynamics was originally derived by Taylor and Jonker [45] in continuous time, describing the relative propagation of the phenotypes in a population where the generation gap $\delta t \to 0$. As the discrete version of the equations is presented below, we may find that a finite generation gap $\delta t > 0$ makes the link with the modelled biological problem even clearer.

$$\dot{p}_i(t) = E(e_i - p(t), p(t))p_i(t), \; i = 1, ..., l, \text{ for } t \geq 0 \tag{5.3}$$

where $p(t) = [p_1(t)...p_l(t)]^T$, $\sum_{i=1}^{l} p_i(t) = 1$,

$$e_i \text{ is the unit vector along axis } i$$

$$\text{and } E(x,y) = x^T G y, \text{ where } G \text{ is an } l \times l \text{ matrix.}$$

In this setting, the variables p_i $i = 1, ..., l$ will be interpreted as the proportions of phenotypes i in the population whereas the matrix G is a fitness matrix where element G_{ij} indicates how well an individual of phenotype i would do in an encounter with an individual of phenotype j.

Deriving the Equations

The continuous equations can be derived [52] in a straightforward way from the simple original assumptions on the growth rate and the large number of population members.

General Case $l < \infty$ Assuming that the initial population at time $t = 0$ is composed of $n(0)$ individuals, we let the number of individuals of phenotype i at any time $t \geq 0$ be denoted $n_i(t)$ and the total number of individuals be $n(t) = \sum_{i=1}^{l} n_i(t)$. If the proportion of individuals of phenotype i at time t is $p_i(t)$, it is thus clear that

$$n_i(t) = n(t)p_i(t). \tag{5.4}$$

Assuming that the relative fitness is measured as the number of offspring per time unit and that $n(t)$ is so large that the weak law of large numbers can be applied, we get equation (5.5), whereas derivation with respect to time on both sides of equation (5.4) gives equation (5.6).

$$\dot{n}_i(t) = E(e_i, p(t))n_i(t) \tag{5.5}$$

$$\dot{n}_i(t) = \dot{n}(t)p_i(t) + n(t)\dot{p}_i(t) \tag{5.6}$$

Since, by equation 5.6, $n(t)\dot{p}_i(t) = \dot{n}_i(t) - \dot{n}(t)p_i(t) = \{\text{eq. } 5.4, 5.5\ \}$

$= E(e_i, p(t))\ n(t)p_i(t) - (\sum_j E(e_j, p(t))\ n_j(t))\ p_i(t)$

$= n(t)E(e_i, p(t))\ p_i(t) - n(t)E(p(t), p(t))\ p_i(t))$, we have

$$n(t)\dot{p}_i(t) = n(t)E(e_i, p(t))p_i(t) - n(t)E(p(t), p(t))p_i(t). \tag{5.7}$$

Dividing by $n(t)$ on both sides, we get the replicator equations:

$$\dot{p}_i(t) = (E(e_i, p(t)) - E(p(t), p(t)))p_i(t), i = 1, ..., l \tag{5.8}$$

Using the abbreviations p_i for $p_i(t)$ and p for $p(t)$, an equivalent expression is for all pairs $\{(i,j) \mid i,j \in \{1,2,...,l\},\ i \neq j\}$:

$$\frac{d}{dt}\frac{p_i}{p_j} = (E(e_i, p) - E(e_j, p))\frac{p_i}{p_j} \tag{5.9}$$

This can be seen since

$$\begin{aligned}\frac{d}{dt}\frac{p_i}{p_j} &= \frac{\dot{p}_i}{p_j} - \frac{p_i}{p_j^2}\dot{p}_j \\ &= \frac{1}{p_j}(E(e_i,p)p_i - E(p,p)p_i) - \frac{p_i}{p_j^2}(E(e_j,p)p_j - E(p,p)p_j) \\ &= (E(e_i,p) - E(e_j,p))\frac{p_i}{p_j}. \qquad (5.10)\end{aligned}$$

Special Case $l = 2$ When there are only two phenotypes, the equations in (5.8) can be simplified as follows since $p_2(t) = 1 - p_1(t)$:

$$\begin{aligned}\dot{p}_1(t) &= (E(e_1,p(t)) - E(p(t),p(t)))p_1(t) = (1-p_1(t))p_1(t)\begin{bmatrix}1 & -1\end{bmatrix} G \begin{bmatrix}p_1(t) \\ 1-p_1(t)\end{bmatrix} \\ &= p_1(t)(1-p_1(t))\Delta_{12}(t), \text{ where } \Delta_{12}(t) = E(e_1 - e_2, p(t)). \qquad (5.11)\end{aligned}$$

Conservation Properties

If the initial values $p_i(0)$, $i = 1,...,l$ satisfy the constraints $0 < p_i(0) < 1$, $i = 1,...,l$ and $\sum_{i=1}^{l} p_i(0) = 1$, then the replicator dynamics will automatically guarantee that the same constraints are satisfied by the variables $p_i(t)$, $i = 1,...,l$ for any $t > 0$. This can be seen since

$$\sum_{i=1}^{l}\dot{p}_i(t) = \sum_{i=1}^{l}(E(e_i,p(t)) - E(p(t),p(t)))p_i(t) = 0 \qquad (5.12)$$

and, if $p_i(t) = 0$ or $p_i(t) = 1$,

$$\dot{p}_i(t) = (E(e_i,p(t)) - E(p(t),p(t)))p_i(t) = 0. \qquad (5.13)$$

Discete Time

The discrete replicator dynamics is given by the following equations:

$$\delta p_i(t) = p_i(t+1) - p_i(t) = \delta t E(e_i - p(t), p(t))p_i(t), \ i = 1,...,l, \qquad (5.14)$$

for $t = j\delta t$, $j = 0,1,2,...$

where $p(t) = [p_1(t)...p_l(t)]^T$, $\sum_{i=1}^{l} p_i(t) = 1$,

e_i is the unit vector along axis i

and $E(x,y) = x^T G y$, where G is an $l \times l$ matrix.

As we go from discrete dynamics to continuous, from a biological point of view there is no longer a synchronized generation gap; likewise, in a game theoretic interpretation, the iterated game is played continuously.

Biological Interpretation

In the original, biological interpretation of the equations, $p(t)$ is the phenotype composition at time t of a population of individuals competing for some scarce resource in pairwise encounters - a phenotype is a distinct version of a given trait relevant to an individual's success in such a competition.

Each individual expresses one and only one phenotype during its entire life span and passes this phenotype on to all its offspring.

The $l \times l$ matrix G indicates the expected outcomes of the pairwise encounters as fitness points - the expected fitness of an individual of phenotype i encountering an individual of phenotype j will thus be given as entry $G_{ij} = e_i^T G e_j$ of matrix G. Likewise, the expected fitness of an individual of phenotype i matched against a randomly picked opponent from the population will be $\sum_{j=1}^{l} e_i^T G e_j p_j(t) = e_i^T G p(t)$.

At each time $t = j\delta t$, $j = 0, 1, 2, \ldots$, the subpopulations represented by the different phenotypes replicate at a rate proportional to their fitness relative to the average fitness. The replication is thus synchronized and occurs in discrete time with a generation gap δt.

Game Theoretic Interpretation

The replicator equations can also be interpreted as the strategy dynamics of an iterated game where, in fact, there is only one collective player - the population - but where each individual faces an infinite number of potential opponents chosen at random - all the other population members.

Each phenotype can be interpreted as a pure strategy - from a game theoretic point of view, the number l of phenotypes thus corresponds to a finite action space of actions $i = 1, 2, \ldots, l$.

The population composition or state $p(t)$ at time t can be seen as a collective strategy played by the population as a whole; from the perspective of each individual population member who will be randomly matched against any other member of the population, this is the expected opponent strategy.

The fitness matrix G is now instead seen as a payoff matrix.

From a game theoretic point of view, all individuals are thus fixed strategists, always playing the same pure strategy or, in some studies, the same mixed strategy. This means that the system is non-adaptive at the individual level and adaptive only at the collective level.

5.2.2 Multi-Population Replicator Dynamics

In the multi-population version of the replicator dynamics, the iterative game is an N-player game played by N populations [52]. Whereas in the single-population replicator dynamics, there was only one collective strategy $p(t)$, the analogy between populations and players is here more straightforward since there are now not only one but $N > 1$ players or populations with their respective population states or strategies $p^k(t)$, $k = 1, ..., N$.

General Multi-Population Case

Introducing the notation $p^{-k}(t)$ for the strategy vector excluding strategy k,

$$p^{-k}(t) = \begin{bmatrix} p^1(t) & p^2(t) & ... & p^{k-1}(t) & p^{k+1}(t) & ... & p^N(t) \end{bmatrix} \tag{5.15}$$

the N-population replicator dynamics can be written as

$$\dot{p}_i^k(t) = (E(e_i, p^{-k}(t)) - E(p^k(t), p^{-k}(t)))p_i^k(t), i = 1, ..., l \tag{5.16}$$

$$\text{for } t \geq 0,\ i = 1, ..., l \text{ and } k = 1, ..., N,$$

$$\text{where } p^k(t) = [p_1^k(t)...p_l^k(t)]^T,\ \textstyle\sum_{i=1}^l p^k(t) = 1,$$

$$e_i \text{ is the unit vector along axis } i$$

$$\text{and } E(p^1(t), p^{-1}(t)) = \textstyle\sum_{i_1=1}^l \cdots \sum_{i_N=1}^l G_{i_1...i_N} p_{i_1}^1(t) p_{i_2}^2(t) ... p_{i_N}^N(t),$$

$$\text{where } G \text{ is an } l^N \text{ matrix.}$$

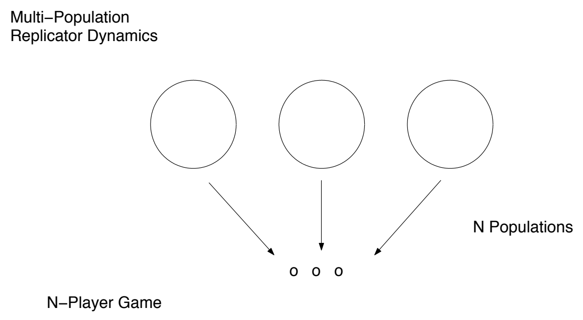

Figure 5.2: The multi-population replicator dynamics.

Two-Population Case

A special case is when $N = 2$, which can be interpreted as individuals from either population being matched only with individuals from the other population rather than with other members of their own population. The equations for this case are given below with the notation $p(t) = p^1(t)$ and $q(t) = p^2(t)$.

$$\dot{p}_i(t) = (E(e_i, q(t)) - E(p(t), q(t)))p_i(t), i = 1, ..., l \tag{5.17}$$
$$\dot{q}_i(t) = (E(e_i, p(t)) - E(q(t), p(t)))q_i(t), i = 1, ..., l \text{ for } t \geq 0$$
$$\text{where } p(t) = [p_1(t)...p_l(t)]^T,\ \textstyle\sum_{i=1}^{l} p(t) = 1,$$
$$q(t) = [q_1(t)...q_l(t)]^T,\ \textstyle\sum_{i=1}^{l} q(t) = 1,$$
$$e_i \text{ is the unit vector along axis } i$$
$$\text{and } E(x, y) = x^T G y, \text{ where } G \text{ is an } l \times l \text{ matrix.}$$

Conservation Properties

The multi-population replicator dynamics has the same conservation properties as the single-population version, since

$$\sum_{i=1}^{l} \dot{p}_i^k(t) = \sum_{i=1}^{l} (E(e_i, p^{-k}(t)) - E(p^k(t), p^{-k}(t)))p_i^k(t) = 0 \tag{5.18}$$

and, if $p_i^k(t) = 0$ or $p_i^k(t) = 1$,

$$\dot{p}_i^k(t) = (E(e_i, p^{-k}(t)) - E(p^k(t), p^{-k}(t)))p_i^k(t) = 0. \tag{5.19}$$

Biological Interpretation

The biological interpretation of the multi-population replicator dynamics differs from the single-population version in two respects.

Firstly, it is assumed that all individuals encounter individuals from a different population than their own, that is, the relative propagation of a phenotype in a given population depends on how well the phenotype does relative to one or several exterior populations.

Secondly, individuals are generally no longer matched pairwise but in groups of N individuals, one from each population.

Game Theoretic Interpretation

The analogy between the multi-population replicator dynamics and an iterative game is more straightforward than for the single-population case, since each population $k = 1, ..., N$ can be seen as a player with current strategy $p^k(t)$ taking part in an N-player game rather than one player playing the game against himself.

5.2.3 Equilibrium Properties

The ESS [33] was introduced above as a refinement of the Nash equilibrium - the ESS is the central equilibrium notion in the single-population replicator dynamics since all ESS's can be shown to be attractors in this standard form of the replicator dynamics [15, 52]. This and some other properties of the ESS needed for our further discussion are presented below.

Totally Mixed ESS Unique

As we saw above, a finite game may have no ESS, one ESS or several ESS's; however, if an ESS has full support, it is the unique ESS of the game. To prove this uniqueness property which will be important in our further discussion, the following two lemmas and definitions are needed [15, 52].

Definition: Best Response The best response by player k to a strategy vector $\mathbf{p} = (p^1, ..., p^N)$ is the set of strategies $q^* \in \mathcal{P}(A)$ such that $q^* = \arg\max_q E_k(q; p^1, ..., p^{k-1}, p^{k+1}, ..., p^N)$. From the definition, we see that a Nash equilibrium is a best response to itself.

Definition: Support The support $R(p)$ of a strategy $p \in \mathcal{P}(A)$ is defined as $R(p) = \{i, i = 1, ..., l \; | p_i > 0\}$.

Lemma 1 Any pure strategy included in the support of a best response q^* to a strategy vector $\mathbf{p}$ is also a best response to $\mathbf{p}$ [15].

Proof The proof is by contradiction.
Assuming $E_k(q^*; p^1, ..., p^{k-1}, p^{k+1}, ..., p^N) = \gamma^*$ and that there exists at least one $a \in R(q^*)$ such that $E_k(\bar{p}^a; p^1, ..., p^{k-1}, p^{k+1}, ..., p^N) < \gamma^*$, where $R(\bar{p}^a) = \{a\}$, then one could increase the payoff by modifying q^* so that the probability of choosing action a would be zero. Since γ^* was the maximum payoff, this is a contradiction.

Definition $S(p) \triangleq \{a \in A \mid E(\bar{p}^a, p) = E(p, p)\}$, $R(\bar{p}^a) = \{a\}$.

Assuming p and q are two distinct ESS's of the same game, the following holds:

Lemma 2 $R(p) \not\subseteq S(q)$ and $R(q) \not\subseteq S(p)$ [6, 10, 48]

Proof The proof is by contradiction:
If $R(q) \subset S(p)$, then $E(q, p) = E(p, p)$ by definition.
Since p is an ESS, we must also have $E(p, q) > E(q, q)$.

Since q is also an ESS, either
(i) $E(q, q) > E(p, q)$ or (ii) $E(q, q) = E(p, q)$ and $E(q, p) > E(p, p)$.

We now have:
$E(q,q) \geq E(p,q)$ and $E(q,q) < E(p,q)$, thus contradiction.

This leads to the following theorem:

Theorem 1 A totally mixed ESS is unique [15, 52, 6, 10, 48].

Proof If p is an ESS and totally mixed, $S(p) = A$ by lemma 1. The theorem follows from lemma 2 since if q is any other ESS, we have
$R(q) \nsubseteq S(p) = A$, which is impossible.

5.2.4 Convergence of Replicator Dynamics

What can be said of the replicator dynamics as the time $t \to \infty$? Indeed, at least for $l = 2$ and $N = 2$, both the single-population and the multi-population versions converge, but with qualitatively different behaviors.

Single-Population Replicator Dynamics

As stated above, every ESS is an attractor in the single-population replicator dynamics; furthermore, if a game has a totally mixed ESS, that is the unique attractor. To prove this, the following lemma will be needed.

Lemma 3 If a strategy p is an ESS of a game represented by its game matrix G, there is a neighborhood B of p such that [15]

$$p^T G\pi > \pi^T G\pi \ \forall \pi \in B, \ \pi \neq p \tag{5.20}$$

Proof For each strategy $q \neq p$ in the strategy space $\mathcal{P}(A)$ of probability distributions over the action space A, we let

$$\pi_\epsilon(q) = (1-\epsilon)p + \epsilon q \text{ and define} \tag{5.21}$$

$$\epsilon(q) = \sup\{\epsilon > 0 \mid p^T G\pi_\epsilon(q) > q^T G\pi_\epsilon(q)\} \tag{5.22}$$
$$\text{and } \epsilon^* = \inf\{\epsilon(q) \mid q \in \mathcal{P}(A)\}. \tag{5.23}$$

Since p is an ESS, the set $\epsilon(q)$ is non-empty for any $q \in P(A)$, $q \neq p$.
This can be seen by fixing q and analyzing the difference

$$\Delta = p^T G\pi_\epsilon(q) - q^T G\pi_\epsilon(q) = (1-\epsilon)(p^T Gp - q^T Gp) + \epsilon(p^T Gq - q^T Gq).$$

If $p^T Gp > q^T Gp$, we can choose $\epsilon > 0$ small enough for Δ to be greater than 0.
If $p^T Gp = q^T Gp$, we know that $p^T Gq > q^T Gq$ and Δ will be greater than 0 for any $0 < \epsilon \leq 1$.

Since $\epsilon(q)$ is a continuous function and $\mathcal{P}(A)$ is a compact set, $\epsilon^* > 0$.

We choose

$$\bar{B} = \{\pi_\epsilon(q) | q \in P(A) \text{ and } 0 \leq \epsilon < \epsilon^*\}. \tag{5.24}$$

Since

$$p^T G \pi_\epsilon(q) > q^T G \pi_\epsilon(q) \ \forall \pi_\epsilon(q) \in \bar{B}, \tag{5.25}$$

we have

$$\begin{aligned} p^T G \pi_\epsilon(q) &= (1-\epsilon) p^T G \pi_\epsilon(q) + \epsilon p^T G \pi_\epsilon(q) \\ &> (1-\epsilon) p^T G \pi_\epsilon(q) + \epsilon q^T G \pi_\epsilon(q) = \pi_\epsilon(q)^T G \pi_\epsilon(q). \end{aligned} \tag{5.26}$$

We let B be the neighborhood

$$B = \{\pi \in \bar{B} || p - \pi| < \min_q |p - \pi_{\epsilon^*}(q)|\}. \tag{5.27}$$

Thus, B is a neighborhood of p that satisfies the desired constraint.

We can now prove the following theorem:

Theorem 2 Every ESS is an attractor in the replicator dynamics [52].

Proof We choose a neighborhood B,

$$B = \{q \in P(A) | p^T G q > q^T G q\}; \tag{5.28}$$

we know from lemma 3 that such a neighborhood exists.
Let p be the ESS and choose any $q \in B$.
We will now show that the Kullback-Leibler distance $D(p|q)$ is a Lyapunov function for the ESS in the replicator dynamics [52].

$$D(p|q) \triangleq \sum_{i=1}^{l} p_i \log \frac{p_i}{q_i} \text{ and } D(p|q) \geq 0,\ D(p|q) = 0 \text{ iff } q = p. \tag{5.29}$$

Also,

$$\begin{aligned} \frac{d}{dt} D(p|q) &= \sum_{i=1}^{l} p_i \frac{-\dot{q}_i}{q_i} = -\sum_{i=1}^{l} p_i (E(e_i, q) - E(q, q)) \\ &= -(E(p, q) - E(q, q)) < 0 \text{ for } q \in B \text{ by lemma 3.} \end{aligned} \tag{5.30}$$

As a corollary, we finally obtain that a totally mixed ESS is the unique attractor:

Corollary 1 Every totally mixed ESS is a global attractor in the replicator dynamics [52].

Proof Let p be the ESS and let $q \in \mathcal{P}(A)$ be any other strategy.
Since every pure strategy $a \in R(q)$ is a best response to p, we have

$$e_i^T Gp = \gamma \text{ for some } \gamma \text{ and each } i. \tag{5.31}$$

Thus,

$$p^T Gp = \sum_{i=1}^{l} p_i \gamma = \sum_{i=1}^{l} q_i \gamma = q^T Gp. \tag{5.32}$$

Since p is an ESS, we must have

$$p^T Gq > q^T Gq. \tag{5.33}$$

By theorem 2, p is a global attractor.

Multi-Population Replicator Dynamics

The typical dynamics of the multi-population replicator dynamics is qualitatively different from the single-population dynamics since interior Nash equilibria typically are no longer attractors but saddle points [52]; instead, the strategies converge to asymmetric Nash equilibria on the closure of the strategy simplex - the equilibrium payoffs to the N populations or players will therefore also be asymmetric. In the game of Hawks and Doves described above, the dynamics will converge to the ESS $\mathbf{p}_3$ in the single-population case but to either of the asymmetric Nash equilibria $\mathbf{p}_1$ or $\mathbf{p}_2$ in the two-population case of the multi-population replicator dynamics [52].

Discrete Time vs. Continuous Time

The convergence results derived for the continuous replicator equations carry over almost unaltered for the discrete case; although it is possible to construct degenerate matrices for which the replicator equations will converge in the continuous case but not in the discrete case, convergence in both cases can as a rule be obtained by translation of the matrix.

5.3 Replicator Learning

After this introduction to the replicator dynamics, we will next introduce a randomized generalization of the replicator dynamics which has traits of both the single-population and the multi-population replicator dynamics; this generalization will be referred to as replicator learning. While a learning algorithm in its own right, this algorithm will also be used to solve our original problem. In this scenario, a set of N populations or players play an iterated two-player game where at each step, the two players are selected at random among the N potential players; the players selected obtain payoffs and update their strategies according to the two-population case of the multi-population replicator dynamics. Our main objective is to study the dynamics of the average player strategy as the number of players $N \to \infty$.

5.3.1 N Finite

Definitions

- Each player $k = 1, ..., N$ is associated with a strategy $p^k(t) \in \mathcal{P}(A)$ at any time t.
- $\bar{p}(t, N)$ is the l-vector defined by $\bar{p}(t, N) = \frac{1}{N} \sum \delta(p - p^k(t))$.
- K_t is a random variable denoting the index of the first player chosen to play the game at time t.
- J_t is a random variable denoting the index of the second player chosen to play the game at time t.
- e^{J_t} is a random variable taking its values in the set of unit vectors $\{e_1, e_2, ..., e_l\}$ with probabilities $P(e_i) = p_i^{J_t}(t)$.
- α is a positive constant.

At each time $t = 0, 1, ...,$ the player pair $\{k_t,\ j_t\}$ is chosen at random according to a uniform probability distribution over the set of distinct pairs among N potential players and matched against each other. Each player generates an action according to its current strategy, which is observed by the other player. The players update their strategies as in the multi-population replicator dynamics for the case $N = 2$. The time t is then advanced to $t+1$, two players k_{t+1} and j_{t+1} are chosen and the same update procedure is repeated. We will next describe first the situation where N is finite and then the case where N is infinite.

Individual Dynamics At the individual level, the dynamics is described as follows:

$$\delta p_i^{K_t}(t) = p_i^{K_t}(t+1) - p_i^{K_t}(t) = \alpha E(e_i - p^{K_t}(t), e^{J_t}(t)) p_i^{K_t}(t) \tag{5.34}$$

The expected difference is, $\mathcal{E}$ denoting expectation:

$$\begin{aligned}\mathcal{E}(\delta p^{K_t}) &= \alpha \frac{1}{N}\frac{1}{N-1}\sum_{k=1}^{N}\sum_{j\neq k}\sum_{i=1}^{l}(e_i - p^k(t))^T G e^i p_i^j(t) p_i^k(t)\\ &= \alpha \frac{1}{N}\frac{1}{N-1}\sum_{k=1}^{N}\sum_{j\neq k}(e_i - p^k(t))^T G p^j(t) p_i^k(t), i = 1, ..., l\end{aligned}$$

where $p^{K_t}(t) = [p_1^{K_t}(t) ... p_l^{K_t}(t)]^T$, $\sum_{i=1}^{l} p_i^{K_t}(t) = 1$,

where $p^{J_t}(t) = [p_1^{J_t}(t) ... p_l^{J_t}(t)]^T$, $\sum_{i=1}^{l} p_i^{J_t}(t) = 1$,

e_i is the unit vector along axis i

and $E(x, y) = x^T G y$, where G is an $l \times l$ matrix.

We note that in a different version of the game, where players can also be matched against themselves, the expected difference for the finite case would instead be

$$\mathcal{E}(\delta p^{K_t}) = \alpha \frac{1}{N}\frac{1}{N}\sum_{k=1}^{N}\sum_{j=1}^{N}(e_i - p^k(t))^T G p^j(t) p_i^k(t), i = 1, ..., l$$

Individual Conservation Properties

Each time player k is selected to play the game against another player j, it updates its strategy as if it met a pure strategist population in the two-population replicator dynamics; its strategy will thus always stay within the strategy simplex.

$$\sum_{i=1}^{l} \delta p_i^k(t) = \sum_{i=1}^{l}(E(e_i, e^{J_t}) - E(p^k(t), e^{J_t})) p_i^k(t) = 0 \tag{5.35}$$

and, if $p_i^k(t) = 0$ or $p_i^k(t) = 1$,

$$\delta p_i^k(t) = (E(e_i, e^{J_t}) - E(p^k(t), e^{J_t})) p_i^k(t) = 0. \tag{5.36}$$

Local Feedback

It is possible to introduce local feedback by letting players play the game against themselves, giving the local component

$$\delta p_i^{K_t}(t) = p_i^{K_t}(t+1) - p_i^{K_t}(t) = \alpha E(e_i - p^{K_t}(t), e^{K_t}(t)) p_i^{K_t}(t) \tag{5.37}$$

This converges to the replicator dynamics as $\alpha \to 0$ and will speed up the global convergence but give a different final configuration of the individual players.

Collective Dynamics from equation (5.34), we have that

$$\delta \bar{p}_i(t,N) = \bar{p}_i(t+1,N) - \bar{p}_i(t,N) = P_i(t,N), \tag{5.38}$$

where $P_i(t,N)$ is the random variable

$$\begin{aligned} P_i(t,N) &= \alpha E(e_i - p^{K_t}(t), e^{J_t}(t)) p_i^{K_t}(t) \\ &= \alpha (e_i - p^{K_t}(t))^T G e^{J_t}(t) p_i^{K_t}(t), \ i = 1, ..., l. \end{aligned} \tag{5.39}$$

We now examine the expected values of the first and last terms of the expression in (5.39). For the first expression, we make the following three notes:

Firstly,

$$\begin{aligned} &\mathcal{E}(\alpha e_i^T G e^{J_t} p_i^k(t)) = \mathcal{E}(e_i^T \alpha G (e^{J_t} p_i^k(t))) \\ =& e_i^T \alpha G (\frac{1}{N} \sum_{k=1}^{N} (\frac{1}{N-1} \sum_{j \neq k} p^j(t)) p_i^k(t)) \\ =& e_i^T \alpha G (\frac{1}{N} \sum_{k=1}^{N} (\frac{1}{N} + \frac{1}{N(N-1)}) (\sum_{j=1}^{N} p^j(t) - p^k(t)) p_i^k(t)) \\ =& e_i^T \alpha G (\frac{1}{N} \sum_{k=1}^{N} p_i^k(t)) (\frac{1}{N} \sum_{j=1}^{N} p^j(t)) + O(\frac{1}{N}) \end{aligned} \tag{5.40}$$

Secondly,

$$\begin{aligned} \mathcal{E}(e^{J_t}) &= \frac{1}{N} \sum_{k=1}^{N} (\frac{1}{N-1} \sum_{j \neq k} p^j(t)) \\ =& \frac{1}{N} \frac{1}{N-1} ((N-1) \sum_{k=1}^{N} p^k(t)) = \frac{1}{N} \sum_{k=1}^{N} p^k(t) \end{aligned} \tag{5.41}$$

Thirdly,

$$\mathcal{E}(p_i^k(t)) = \frac{1}{N} \sum_{k=1}^{N} p_i^k(t) \tag{5.42}$$

Thus, we have

$$\mathcal{E}(\alpha e_i^T G e^{J_t} p_i^{K_t}(t)) = \alpha e_i^T G \bar{p}(t,N) \bar{p}_i(t,N) + O(\frac{1}{N}) \tag{5.43}$$

For the second term in (21), we make the following two notes:

Firstly,

$$\mathcal{E}((p^k)^T(t)\alpha G e^{J_t} p_i^k(t)) \tag{5.44}$$
$$=\frac{1}{N}\sum_{k=1}^{N} p_i^k(t)(p^k)^T(t)\alpha G(\frac{1}{N-1}\sum_{j\neq k} p^j(t)) \tag{5.45}$$
$$=\frac{1}{N}\sum_{k=1}^{N} p_i^k(t)(p^k)^T(t)\alpha G((\frac{1}{N}+\frac{1}{N(N-1)})\sum_{j=1}^{N} p^j(t) - p^k(t))$$
$$=(\frac{1}{N}\sum_{k=1}^{N} p_i^k(t)(p^k)^T(t))\alpha G(\frac{1}{N}\sum_{j=1}^{N} p^j(t)) + O(\frac{1}{N})$$

Secondly,

$$\mathcal{E}(e^{J_t}) = \frac{1}{N}\sum_{k=1}^{N} p^k(t) \tag{5.46}$$

Thus,

$$\mathcal{E}(-\alpha(p^{K_t}(t))^T G e^{J_t} p_i^{K_t}(t)) = -\alpha\mathcal{E}((\bar{p}(t,N))^T \bar{p}(t,N)_i)G\bar{p}(t,N) + O(\frac{1}{N}) \tag{5.47}$$

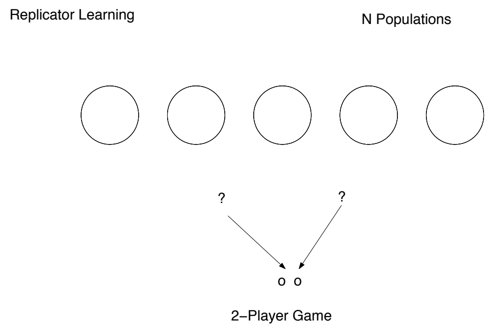

Figure 5.3: Replicator learning.

Collective Conservation Properties

To show the conservation properties for replicator learning, we need the following lemma:

Lemma 4 $\mathcal{E}_{\mu_t}((p_i - \bar{p}_i)(p_i - \bar{p}_i)) = -\sum_{j\neq i}^{l} \mathcal{E}_{\mu_t}((p_i - \bar{p}_i)(p_j - \bar{p}_j))$

Proof

$$\mathcal{E}_{\mu_t}((p_i - \bar{p}_i)(p_i - \bar{p}_i)) = \mathcal{E}_{\mu_t}(p_i p_i) - \bar{p}_i^2(t) \tag{5.48}$$
$$= \mathcal{E}_{\mu_t}(p_i) - \sum_{j\neq i} \mathcal{E}_{\mu_t}(p_i p_j) - \bar{p}_i(t)(1 - \sum_{j\neq i} \bar{p}_j(t))$$
$$= \bar{p}_i - \sum_{j\neq i} \mathcal{E}_{\mu_t}(p_i p_j) - \bar{p}_i + \sum_{j\neq i} \bar{p}_i(t)\bar{p}_j(t) \tag{5.49}$$
$$= -\sum_{j\neq i} \mathcal{E}_{\mu_t}((p_i - \bar{p}_i)(p_j - \bar{p}_j)).$$

We can now show

$$\sum_{i=1}^{l} \dot{\bar{p}}_i(t) = \sum_{i=1}^{l} (E(e_i, \bar{p}(t)) - E(\bar{p}(t), \bar{p}(t)))\bar{p}_i(t)$$
$$-\sum_{i=1}^{l}\sum_{j=1}^{l} \mathcal{E}_{\mu_t}((p_i - \bar{p}_i)(p_j - \bar{p}_j))E(e_j, \bar{p}(t))$$
$$= 0 + \sum_{j=1}^{l}\sum_{i=1}^{l} \mathcal{E}_{\mu_t}((p_i - \bar{p}_i)(p_j - \bar{p}_j))E(e_i, \bar{p}(t))$$
$$= \{\text{Lemma 4}\} = 0 \tag{5.50}$$
$$\tag{5.51}$$

Also, if $\bar{p}_i(t) = 0$ or $\bar{p}_i(t) = 1$, $\mathcal{E}_{\mu_t}((p_i - \bar{p}_i)(p_j - \bar{p}_j)) = 0 \; \forall i, j$, making

$$\dot{\bar{p}}_i(t) = (E(e_i, \bar{p}(t)) - E(\bar{p}(t), \bar{p}(t)))\bar{p}_i(t) = 0. \tag{5.52}$$

Thus, the collective strategy always stays within the strategy simplex.

We are interested in the case where N is large, which implies that α must be small; it is therefore of interest to study the case where N is infinite and $\alpha \to 0$. However, let us first establish how results obtained for infinite N link to our original scenario with random interactions in discrete time.

Theorem 3 For each $\delta > 0$, $\exists\ \alpha > 0$ sufficiently small and N sufficiently large so that, if q is the unique ESS,

$$\lim_{t\to\infty} \sup \frac{1}{t} \sum_{\tau=0}^{t-1} \mathcal{E}(|q - \bar{p}(\tau, N)|^2) \leq \delta \tag{5.53}$$

Proof For each $t = 0, 1, 2, ...,$ let

$$L_t = D(q|\bar{p}(t, N)). \tag{5.54}$$

Then

$$\begin{aligned} L_{t+1} - L_t &= \sum_i q_i \log \frac{\bar{p}_i(t, N)}{\bar{p}_i(t+1, N)} \\ &= -\sum_i q_i \log \frac{\bar{p}_i(t+1, N)}{\bar{p}_i(t, N)} \\ &= -\sum_i q_i \log(1 + \alpha\Delta_i(t, N)) \end{aligned} \tag{5.55}$$

where $\{\Delta_i(t, N)\}$ is a random process defined as

$$\Delta_i(t, N) = \frac{1}{\bar{p}(t, N)} \frac{1}{N} \sum_{k=1}^{N} p_i^k(t)(e_i - p^k(t))^T G e^{J_t} I_{K_t = k} \tag{5.56}$$

Since $\exists\ C_0 < \infty$ such that $\Delta_i(t, N) < C_0$, $\exists\ C_1 > 0$ such that

$$-\log(1 + \alpha\Delta_i(t, N)) \leq -\alpha\Delta_i(t, N) + C_1\alpha^2 \tag{5.57}$$

Thus,

$$\mathcal{E}(L_{t+1} - L_t | p^k(t), k = 1, ..., N) \leq -\alpha \sum_i q_i \Delta_i(t, N) + C_1\alpha^2 \tag{5.58}$$

We have

$$\begin{aligned} &\mathcal{E}(\Delta_i(t, N) | p_i^k(t), i = 1, ..., l, k = 1, ..., N) \\ &= \frac{1}{N(N-1)} \sum_k \sum_{j \neq k} \frac{1}{\sum_{k=1}^{N} p_i^k(t)} p_i^k(t)(e_i - p^k(t))^T G p^j(t) \\ &= \frac{1}{N}(e_i - \bar{p}(t))^T G \bar{p}(t, N) + \epsilon_i^1(t) + \frac{1}{N}\epsilon_i^2(t)) \end{aligned} \tag{5.59}$$

for appropriately defined bounded processes $\epsilon_i^1(t)$ and $\epsilon_i^2(t)$.

Thus, substituting this in the above equation, we have

$$\begin{aligned}&\mathcal{E}(L_{t+1}-L_t|p^k(t),k=1,...,N)\le C_1\alpha^2\\ =&\frac{\alpha}{N}((q-\bar{p}(t,N))^TG\bar{p}(t,N)+\sum_i q_i\epsilon_i^1(t)+\frac{C_1}{N})\end{aligned} \tag{5.60}$$

If q is the unique asymptotically stable ESS of G, we have a constant $C_2>0$ such that for all $\xi=(\xi_1,...,\xi_l)$ such that $\sum_i \xi_i=0$, we have

$$\xi^TGq=0 \text{ and } \xi^TG\xi\le -C_2|\xi|^2 \tag{5.61}$$

Furthermore, under additional assumptions on G, namely that G can be written as

$$G=\begin{bmatrix} a_1 & 0 & 0 & 2a_l\\ 0 & a_2 & 0 & 2a_l\\ \ddots & 0 & \ddots & 2a_l\\ 0 & \ddots & 0 & a_l\end{bmatrix}$$

where $a_i<0, i=1,...,l-1$ and $a_l>0$, we have $\sum_i q_i\epsilon_i^q(t)\le 0$. Thus,

$$\mathcal{E}(L_{t+1}-L_t|p^k(t),k=1,...,N)\le C_1\alpha^2-\frac{\alpha}{N}(C_2|q-\bar{p}(t,N)|^2-\frac{C_1}{N}). \tag{5.64}$$

Taking expectations on both sides of this inquality gives

$$\mathcal{E}(L_{t+1})-\mathcal{E}(L_t)\le C_1\alpha(\alpha+\frac{1}{N^2})-C_2\frac{\alpha}{N}\mathcal{E}(|q-\bar{p}(t,N)|^2). \tag{5.65}$$

Since $L_t\ge 0$ for all t, we have

$$0\le\mathcal{E}(L(t))\le\mathcal{E}(L_0)+tC_1\alpha(\alpha+\frac{1}{N^2})=C_2\frac{\alpha}{N}\sum_{m=1}^{t-1}\mathcal{E}(|q-\bar{p}(t,N)|^2) \tag{5.66}$$

Thus,

$$\frac{1}{t}\sum_{\tau=0}^{t=1}\mathcal{E}(q-\bar{p}(\tau,N)|^2)\le\frac{1}{t}\mathcal{E}(L_0)+(N\alpha+\frac{1}{N}) \tag{5.67}$$

Thus, if the number of players is large enough and the step size small enough, then a stability result from the continuous time case with infinite N carries over to our original scenario with random interactions in discrete time.

5.3.2 N Infinite

Definitions

- The random variable μ_t: $\Omega \to \mathcal{P}(\mathcal{P}(A))$ denotes the probability that a randomly chosen player has a certain strategy and takes its values in the space of probability measures over $\mathcal{P}(A)$; $\mu_t^{N,\alpha}$ is its equivalent when N is finite and α non-infinitesimal.
- $\bar{p}(t)$ is the l-vector defined by $\bar{p}_i(t) = \int p_i d\mu_t$, $i = 1, ..., l$.

We next study the convergence of the process $\{p^k(t), k = 1, ..., N\}$ as $\alpha \to 0$ and $N \to \infty$.

We first study the limit $\alpha \to 0$. The following result is the consequence of a standard result from stochastic approximation literature, see theorem 1 on p.101 in [29]. To state the theorem we introduce the linearly interpolated process $\{p^k(t); k = 1, ..., N\}$ defined as

$$p^k_\alpha(t) = (\frac{Nt}{\alpha} - [\frac{Nt}{\alpha}])p^k([\frac{Nt}{\alpha}] + 1) + (1 - \frac{Nt}{\alpha} + [\frac{Nt}{\alpha}])p^k([\frac{Nt}{\alpha}]) \tag{5.68}$$

Theorem 4.1 The family of probability measures on $C([0,\infty); \mathcal{P}(A)^N)$ (with the topology of uniform convergence) corresponding to the laws of $\{p^k(t); k = 1, ..., N\}$ parametrized by α is tight and every limit point of this set as $\alpha \to 0$ has its support on the solutions of ODE

$$\dot{p}^k = \frac{1}{N-1} \sum_{j \neq k} p_i^k (e_i - p^k)^T G p^j \tag{5.69}$$

The theorem is applicable since the noise - p^j in this case - does not depend on the state p^k and since if $|G| < \infty$,

(i) for each $\tau < \infty$, $p_i^k(e_i - p^k)^T G p^j$ is a polynomial and the set

$$\{\sup_{p^k} |p_i^k(m)(e_i - p^k(m))^T G p^j(m)|,\ \alpha > 0,\ m \geq 0,\ m\alpha \leq \tau\}$$

is uniformly integrable.

(ii) for each random variable $p \in \mathcal{P}(A)$ and each $\tau < \infty$, since $p_i^k(e_i - p^k)^T G p^j$ is continuous

$$\lim_{m \to \infty, \delta \to 0, \alpha \to 0} \mathcal{E} \sup_{|\pi| \leq \delta} |p_i(e_i - p)^T G p^j - (p_i + \pi_i)(e_i - (p + \pi))^T G p^j|$$
$$= \lim_{m \to \infty, \delta \to 0, \alpha \to 0} \mathcal{E} \sup_{|\pi| \leq \delta} |p_i(e_i - p)^T G p^j - p_i(e_i - p)^T G p^j + O(|\pi|)| \to 0$$

(iii) there are $\{n_\alpha\}$ and a continuous function $f(\cdot)$ such that $n_\alpha \to \infty$ and $\delta_\alpha \triangleq \alpha n_\alpha \to 0$ and for each p we have as $\alpha \to 0$ and $n \to \infty$

$$\tfrac{1}{n_\alpha} \textstyle\sum_{m=n}^{n+n_\alpha-1} \mathcal{E}_n^\alpha (p_i^k(m)(e_i - p^k(m))^T G p^j(m)) \to f(p);$$

this is a weakened form of the weak law of large numbers.

We now want to study this family of ODE's as $N \to \infty$. Since the dimension of the ODE goes to infinity as $N \to \infty$, we define a trajectory in $\mathcal{P}(\mathcal{P}(A)$ associated with each N as follows:

$$\mu_t^N(B) = \frac{1}{N} \sum_k I_{p^k(t) \in B}. \tag{5.70}$$

For each subset B of mixed strategies, $\mu_t^{N,\alpha}(B)$ denotes the proportion of the players using a strategy in B at time t. By first letting $\alpha \to 0$ and then $N \to \infty$, we avoid the difficulty of assuring the tightness of a sequence in an infinite dimensional space.

Theorem 4.2 The family of functions $\{\mu_t^N; N = 1, ..,\}$ in $C([0,\infty); \mathcal{P}(\mathcal{P}(A))$ (with the topology of uniform convergence on compact intervals) is relatively compact. Furthermore, any limit point of these functions μ_t satisfies the ODE in weak form - for each $f \in C_b^\infty(\mathcal{P}(A))$,

$$\frac{d}{dt} < f, \mu_t >= \int \sum_i \frac{\partial f}{\partial p_i} p_i(e_i - p)^T G \int q \mu_t(dq) \mu_t(dp) \tag{5.71}$$

Proof The relative compactness of $\{\mu_t^N; N = 1, ...\}$ follows from Ascoli theorem [30].
Fix an $f \in C_b^\infty(\mathcal{P}(A))$. Then, for any $t > 0$, we have

$$< f, \mu_t^N > - < f, \mu_0^N >= \frac{1}{N} \sum_k (f(p^k(t)) - f(p^k(0))). \tag{5.72}$$

For each function μ_t in $C([0,\infty); \mathcal{P}(\mathcal{P}(A))$, define another function in $C([0,\infty); \mathcal{P}(\mathcal{P}(A))$

$$\epsilon_t(\mu_t) =< f, \mu_t >=< f, \mu_0^{N,\alpha} > - \int_0^t \sum_i \frac{\partial f}{\partial p_i} p_i(e_i - p)^T G \int q \mu_s(dq) \mu_s(dp) ds. \tag{5.73}$$

To prove the result, it is enough to prove that for any $T > 0$

$$\lim_{N \to \infty} \sup \sup_{t \leq T} \epsilon_t |(\mu_t^N)| = 0 \tag{5.74}$$

This is easy to prove since the right hand sides of equations (5.69) and (5.72) are O($\frac{1}{N}$).

Finally, we can derive the dynamics of $\bar{p}(t)$ as follows:

$$\begin{aligned}
&\text{Since } \dot{\bar{p}}_i(t) = \frac{d}{dt} < \mu_t, p_i > \text{ and} \\
&\int [E(e_i - p, \bar{\mu}_t(p))p_i] d\bar{\mu}_t = \int \sum_j \sum_k (\delta_{ij} - p) p_i G_{jk} \bar{p}_k(t) d\bar{\mu}_t \\
=& \int \sum_k p_i G_{jk} \bar{p}_k(t) d\bar{\mu}_t - \int \sum_j \sum_k p_j p_i G_{jk} \bar{p}_k(t) d\bar{\mu}_t \\
=& \sum_k \bar{p}_i(t) G_{jk} \bar{p}_k(t) - \sum_j \sum_k \bar{p}_i(t) \bar{p}_j(t) G_{jk} \bar{p}_k(t) \\
-& \sum_j \sum_k \mathcal{E}_{\mu_t}(p_i - \bar{p}_i(t))(p_j - \bar{p}_j(t)) G_{jk} \bar{p}_k(t) \\
=& E(e_i - \bar{p}(t), \bar{p}(t)) - \sum_j \mathcal{E}_{\mu_t}(p_i - \bar{p}_i(t))(p_j - \bar{p}_j(t)) E(e_j, \bar{p}(t))
\end{aligned}$$

$$i = 1, ..., l, \text{ for } t \geq 0 \tag{5.75}$$

where $\bar{p}(t) = [\bar{p}_1(t)...\bar{p}_l(t)]^T$, $\sum_{i=1}^l \bar{p}_i(t) = 1$,

e_i is the unit vector along axis i,

and $E(x, y) = x^T G y$, where G is an $l \times l$ matrix.

5.3.3 Variance

Remembering the original desire to spread out the units as much as possible in a search scenario, we are furthermore interested in the dynamics of the variance of μ_t, expressed as

$$\begin{aligned}
\frac{d}{dt} < \mu_t, (p_i - \bar{p}_i)^2 > &= < \mu_t, L(p_i^2) > - 2\bar{p}_i \dot{\bar{p}}_i \\
&= \mathcal{E}_{\mu_t}(2E(e_i - p, \bar{p}(t)) p_i^2) - 2\bar{p}_i \dot{\bar{p}}_i
\end{aligned} \tag{5.76}$$

In the replicator dynamics, the individual units are static pure strategists which can be found at the corners of the simplex. In replicator learning, the individual units are instead adaptive and move within the simplex; it would be interesting to study the stability of the above equation for the evolution of the variance, since in a search application one typically wishes this variance to satisfy certain constraints, given an estimate of the probability of finding the lost object in different areas.

5.3.4 Replicator Learning vs. Replicator Dynamics

Before studying the stability, we will briefly compare replicator learning with the replicator dynamics to make clear what is similar and what is different.

Different Matching As stated above, the matching used in replicator learning is different from both the single-population replicator dynamics and the general multi-population dynamics; it is best described as a randomized version of the two-population multi-population dynamics.

Two Adaptive Levels In the single-population and multi-population versions of the replicator dynamics, the individuals are fixed strategists whereas the collective strategies $p^k(t)$, $k = 1, ..., N$ are adaptive. Thus, in the replicator dynamics, the system is static at the individual level and adaptive only at the collective level.

In replicator learning, we define a superstructure in the form of the average strategy $\bar{p}(t)$, just like the phenotype composition of a population was determined as the average phenotype of the individuals seen as fixed and pure strategists. In this way, we obtain a two-layered framework with adaptivity in both layers rather than in just one.

Variance The replicator learning dynamics will thus vary with the statistics of the node positions. To illustrate this, we examine the two extreme situations when the variance is maximal or minimal - in the former case, the dynamics vanishes whereas in the latter case, the replicator learning dynamics reduces to the replicator dynamics.

Maximal Variance If the variance is maximal and $\bar{p}(t)$ is no longer required to be an interior point, the dynamics vanishes; however, we always require $0 < p_i^k(t) < 0$ which in turn ensures $0 < \bar{p}(t) < 1$.

For any i, $\mathcal{E}_{\mu_t}((p_i - \bar{p}_i)^2) \leq \bar{p}_i(1 - \bar{p}_i)$.
Also, for any i, $\mathcal{E}_{\mu_t}((p_i - \bar{p}_i)^2) = -\sum_{j \neq i} \mathcal{E}_{\mu_t}((p_i - \bar{p}_i)(p_j - \bar{p}_j))$.

$$\text{If } \mathcal{E}_{\mu_t}((p_i - \bar{p}_i)^2) = \bar{p}_i(1 - \bar{p}_i), \text{then } \mathcal{E}_{\mu_t}((p_i - \bar{p}_i)(p_j - \bar{p}_j)) = -\bar{p}_j\bar{p}_i \ \forall j \neq i.$$

Thus, if $\mathcal{E}_{\mu_t}((p_i - \bar{p}_i)^2) \leq \bar{p}_i(1 - \bar{p}_i)$, we have:

$$\begin{aligned}
&\sum_j \mathcal{E}_{\mu_t}((p_i - \bar{p}_i)(p_j - \bar{p}_j))\alpha e_j^T G\bar{p} \\
=&\sum_{j \neq i} \mathcal{E}_{\mu_t}((p_i - \bar{p}_i)(p_j - \bar{p}_j))\alpha e_j^T G\bar{p} + \mathcal{E}_{\mu_t}((p_i - \bar{p}_i)^2)\alpha e_i^T G\bar{p}
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{j\neq i} \mathcal{E}_{\mu_t}((p_i - \bar{p}_i)(p_j - \bar{p}_j))\alpha e_j^T G\bar{p} - \sum_{j\neq i} \mathcal{E}_{\mu_t}((p_i - \bar{p}_i)(p_j - \bar{p}_j))\alpha e_i^T G\bar{p} \\
&= \sum_{j\neq i} \mathcal{E}_{\mu_t}((p_i - \bar{p}_i)(p_j - \bar{p}_j))(\alpha e_j^T G\bar{p} - \alpha e_i^T G\bar{p}) \\
&= \sum_{j\neq i} (-\bar{p}_j\bar{p}_i)(\alpha e_j^T G\bar{p} - \alpha e_i^T G\bar{p}) \\
&= \sum_{j=1}^{l} (-\bar{p}_i\bar{p}_j)(\alpha e_j^T G\bar{p} - \alpha e_i^T G\bar{p}) \\
&= \bar{p}_i(\sum_{j=1}^{l}(-\bar{p}_j)\alpha e_j^T G\bar{p}) + \bar{p}_i(\sum_{j=1}^{l}\bar{p}_j \alpha e_i^T G\bar{p}) \\
&= \bar{p}_i(-\bar{p}^T\alpha G\bar{p} + \bar{p}_i e_i^T \alpha G\bar{p}) \\
&= e_i^T\alpha G\bar{p}\bar{p}_i - \bar{p}^T\alpha G\bar{p}\bar{p}_i,
\end{aligned}
$$

which would make the expression in (31) zero.

Minimal Variance When the variance is zero, the second term in the differential equation for replicator learning vanishes, leaving as a result the replicator dynamics:

If $\mathcal{E}_{\mu_t}((p_i - \bar{p}_i)^2) = 0$, then $p_i^k = p_i \ \forall k$ and we thus have $\forall j \neq i$:

$$\mathcal{E}_{\mu_t}((p_i - \bar{p}_i)(p_j - \bar{p}_j)) = \mathcal{E}_{\mu_t}(p_i p_j) - \bar{p}_j\bar{p}_i = \mathcal{E}_{\mu_t}(p_j)\bar{p}_i - \bar{p}_i\bar{p}_j = \bar{p}_i\bar{p}_j - \bar{p}_i\bar{p}_j = 0.$$

Alternative Formulae The replicator dynamics is as a rule encountered in its standard formula as seen above (5.3); the replicator learning equations introduced above generalize these equations by adding a second term to the differential equations. However, to express the simliarities and differences between the replicator dynamics and replicator learning in an even more compact way, we will derive the following alternative formulae, which will be used below in the stability proof for the general case when $l > 2$:

$$\delta\bar{p}_i = \sum_{j\neq i} \bar{p}_i\bar{p}_j\Delta_{ij}, \text{ where } \Delta_{ij} = \alpha e_i^T G\bar{p} - \alpha e_j^T G\bar{p} \text{ and} \tag{5.79}$$

$$\delta\bar{p}_i = \sum_{j\neq i} \mathcal{E}_{\mu_t}(p_i p_j)\Delta_{ij}, \text{ where } \Delta_{ij} = \alpha e_i^T G\bar{p} - \alpha e_j^T G\bar{p} \tag{5.80}$$

These formulae are derived as follows for the replicator dynamics:

$$\begin{aligned}
\delta\bar{p}_i &= (e_i - \bar{p})^T \alpha G\bar{p}\bar{p}_i = \bar{p}_i \alpha e_i^T G\bar{p} - \sum_j \bar{p}_i\bar{p}_j \alpha e_j^T G\bar{p} \qquad (5.81)\\
&= \bar{p}_i(\bar{p}_1 + ... + \bar{p}_l)\alpha e_i^T G\bar{p} - \sum_j \bar{p}_i\bar{p}_j\alpha e_j^T G\bar{p}\\
&= \bar{p}_i^2 \alpha e_i^T G\bar{p} - \bar{p}_i^2 \alpha e_i^T G\bar{p} + \sum_{j\neq i} \bar{p}_i\bar{p}_j(\alpha e_i^T G\bar{p} - \alpha e_j^T G\bar{p})\\
&= \sum_{j\neq i} \bar{p}_i\bar{p}_j(\alpha e_i^T G\bar{p} - \alpha e_j^T G\bar{p}) = \sum_{j\neq i} \bar{p}_i\bar{p}_j\Delta_{ij}
\end{aligned}$$

And as follows for replicator learning:

$$\begin{aligned}
\delta\bar{p}_i &= (e_i - \bar{p})^T \alpha G\bar{p}\bar{p}_i - \sum_j \mathcal{E}_{\mu_t}((p_i - \bar{p}_i)(p_j - \bar{p}_j))\alpha e_j^T G\bar{p} \qquad (5.82)\\
&= \bar{p}_i \alpha e_i^T G\bar{p} - \sum_j \bar{p}_i\bar{p}_j(\alpha G\bar{p})_j - \sum_j \mathcal{E}_{\mu_t}((p_i - \bar{p}_i)(p_j - \bar{p}_j))(\alpha e_j^T Gp)\\
&= \bar{p}_i \alpha e_i^T G\bar{p} - \sum_j \mathcal{E}_{\mu_t}((p_i - \bar{p}_i)(p_j - \bar{p}_j)) + \bar{p}_i\bar{p}_j)\alpha e_j^T G\bar{p}\\
&= \bar{p}_i \alpha e_i^T G\bar{p} - \sum_j \mathcal{E}_{\mu_t}(p_ip_j)\alpha e_j^T G\bar{p}\\
&= \bar{p}_i(\bar{p}_1 + ... + \bar{p}_l)\alpha e_i^T G\bar{p} - \sum_j \mathcal{E}_{\mu_t}(p_ip_j)\alpha e_j^T G\bar{p}\\
&= (\bar{p}_i^2 - \mathcal{E}_{\mu_t}(p_i^2))\alpha e_i^T G\bar{p} + \sum_{j\neq i} \bar{p}_i\bar{p}_j\alpha e_i^T G\bar{p} - \sum_{j\neq i} \mathcal{E}_{\mu_t}(p_ip_j)\alpha e_j^T G\bar{p}\\
&= -\mathcal{E}_{\mu_t}((p_i - \bar{p}_i)^2)\alpha e_i^T G\bar{p} + \sum_{j\neq i} \bar{p}_i\bar{p}_j\alpha e_i^T G\bar{p} - \sum_{j\neq i} \mathcal{E}_{\mu_t}(p_ip_j)\alpha e_j^T G\bar{p}\\
&= \sum_{j\neq i} \mathcal{E}_{\mu_t}((p_i - \bar{p}_i)(p_j - \bar{p}_j))\alpha e_i^T Gp + \sum_{j\neq i} \bar{p}_i\bar{p}_j\alpha e_i^T Gp - \sum_{j\neq i} \mathcal{E}(p_ip_j)\alpha e_j^T G\bar{p}\\
&= \sum_{j\neq i} \mathcal{E}_{\mu_t}((p_i - \bar{p}_i)(p_j - \bar{p}_j)) + \bar{p}_i\bar{p}_j)\alpha e_i^T Gp - \sum_{j\neq i} \mathcal{E}_{\mu_t}(p_ip_j)\alpha e_j^T Gp\\
&= \sum_{j\neq i} \mathcal{E}_{\mu_t}(p_ip_j)\alpha e_i^T G\bar{p} - \alpha e_j^T G\bar{p} = \sum_{j\neq i} \mathcal{E}_{\mu_t}(p_ip_j)\Delta_{ij}
\end{aligned}$$

As we see, the replicator learning dynamics is similar to the replicator dynamics in so far as the expected product $\mathcal{E}_{\mu_t}(p_ip_j)$ is similar to the product of expectations $\bar{p}_i\bar{p}_j$. In general, $\mathcal{E}_{\mu_t}(p_ip_j)$ will be smaller than $\bar{p}_i\bar{p}_j$, making the dynamics of $\bar{p}_i$ somewhat faster for the replicator dynamics, but if both $\bar{p}_i$ and $\bar{p}_j$ are quite small, it is possible that $\mathcal{E}_{\mu_t}(p_ip_j)$ is instead the larger of the two products, thus instead giving faster dynamics with replicator learning.

5.3.5 Convergence of Collective Dynamics

In this section, we will first look specifically at the case $l = 2$ and find that the Lyapunov function used to prove stability in the single-population replicator dynamics is indeed a Lyapunov function for the replicator learning dynamics as well. We will then show that for a particular choice of the game matrix G, this is also true for the general case if G has a totally mixed ESS.

Theorem 5.1 Every ESS q is an attractor in the replicator learning dynamics for $l = 2$.

Proof The Kullback-Leibler function $D(q|p(t))$,

$$D(q|\bar{p}(t)) \triangleq \sum_{i=1}^{2} q_i \log \frac{q_i}{\bar{p}_i(t)} \tag{5.83}$$

is a Lyapunov function, since

$D(q|\bar{p}(t)) \geq 0,\ D(q|\bar{p}(t)) = 0$ iff $\bar{p}(t) = q$

and since, introducing the notation $\Delta_{ij}(t) = E(e_i - e_j, \bar{p}(t))$,
we have

$$\frac{d}{dt} D(q|\bar{p}(t)) = \sum_{i=1}^{2} q_i \frac{-\dot{\bar{p}}_i(t)}{\bar{p}_i(t)}$$

$$= -\sum_{i=1}^{2} \frac{q_i}{\bar{p}_i(t)} (((E(e_i, \bar{p}(t)) - E(\bar{p}(t), \bar{p}(t)))\bar{p}_i(t) - \sum_{j=1}^{2} \mathcal{E}_{\mu_t}((p_i - \bar{p}_i)(p_j - \bar{p}_j))E(e_j, \bar{p}(t)))$$

$=$\{equation (5.11)\}

$$= -\frac{q_1}{\bar{p}_1(t)} (\bar{p}_1(t)(1 - \bar{p}_1(t))\Delta_{12}(t)) + \frac{(1 - q_1)}{(1 - \bar{p}_1(t))} (\bar{p}_1(t)(1 - \bar{p}_1(t))\Delta_{12}(t))$$

$$-\frac{q_1}{\bar{p}_1(t)} \mathcal{E}_{\mu_t}((p_1 - \bar{p}_1)(p_1 - \bar{p}_1))\Delta_{12}(t) + \frac{(1 - q_1)}{(1 - \bar{p}_1(t))} \mathcal{E}_{\mu_t}((p_1 - \bar{p}_1)(p_1 - \bar{p}_1))\Delta_{12}(t)$$

$$= -\, q_1(1 - \bar{p}_1(t))\Delta_{12}(t) + (1 - q_1)\bar{p}_1(t)\Delta_{12}(t)$$

$$-\frac{\mathcal{E}_{\mu_t}((p_1-\bar{p}_1)(p_1-\bar{p}_1))\Delta_{12}(t)}{\bar{p}_1(t)(1-\bar{p}_1(t))}(q_1(1-\bar{p}_1(t))-\bar{p}_1(t)(1-q_1))$$

$$=(1-\frac{\mathcal{E}_{\mu_t}((p_1-\bar{p}_1)(p_1-\bar{p}_1))}{p_1(t)(1-p_1(t))})\Delta_{12}(t)(\bar{p}_1(t)-q_1)$$

$$=\{\mathcal{E}_{\mu_t}((p_1-\bar{p}_1)(p_1-\bar{p}_1))<\bar{p}_1(t)(1-\bar{p}_1(t))\}$$

$$=|1-\frac{\mathcal{E}_{\mu_t}((p_1-\bar{p}_1)(p_1-\bar{p}_1))}{\bar{p}_1(t)(1-\bar{p}_1(t))}|\Delta_{12}(t)(\bar{p}_1(t)-q_1)=\{\text{Theorem 2}\}<0 \tag{5.84}$$

Theorem 5.2 For any totally mixed $q \in \mathcal{P}(A)$, $|A| = l < \infty$, there is an $l \times l$ matrix G such that q is an ESS of G and such that q is the global attractor in replicator learning.

Proof For each $q \in \mathcal{P}(A)$, $|A| = l$, there is an infinite number of $l \times l$ matrices G that have q as their unique ESS; we know that the set of ESS's of a matrix is invariant to the addition of the same constant c_{add} to each matrix element and to the multiplication of each matrix element by the same positive constant c_{mult}. For each $l < \infty$, we have $l(l-1)$ degrees of freedom in choosing G; we now fix the matrix pattern as follows:

$$G = \begin{bmatrix} a_1 & 0 & 0 & 2a_l \\ 0 & a_2 & 0 & 2a_l \\ \ddots & 0 & \ddots & 2a_l \\ 0 & \ddots & 0 & a_l \end{bmatrix} \tag{5.85}$$

The elements a_i are a function of the desired unique ESS q. For G to be diagonally dominated (5.33), we have $a_i < 0$, $i = 1, ..., l-1$ and $a_l > 0$. We have (5.31)

$$a_i q_i + 2a_l q_l = a_l q_l \Rightarrow q_i = -\frac{a_l}{a_i} q_l \tag{5.86}$$

Since q is a probability distribution, we have

$$q_l(1-\sum_{i=1}^{l-1}\frac{a_l}{a_i}) = 1 \tag{5.87}$$

The ESS can now be obtained as a function of a_i as follows:

$$q_i = -\frac{\Pi_{j\neq i} a_j}{\Pi_{j\neq l} a_j - \sum_{m=1}^{l-1} \Pi_{j\neq m} a_j}, i \neq l$$

$$q_l = \frac{\Pi_{j\neq l} a_j}{\Pi_{j\neq l} a_j - \sum_{m=1}^{l-1} \Pi_{j\neq m} a_j} \tag{5.88}$$

$$\tag{5.89}$$

The time derivative of the Kullback-Leibler function $\frac{d}{dt}D(q|\bar{p}(t))$ can be expressed as a linear function of $\Delta_{ij}(q_j\bar{p}_i(t) - q_i\bar{p}_j(t))$ with positive coefficients whether $\bar{p}(t)$ evolves according to the replicator dynamics or according to replicator learning, as we will see next. Making use of our alternative formulae for the replicator dynamics (5.81) and replicator learning (5.82), respectively, and letting $f_{ij} = \bar{p}_i(t)\bar{p}_j(t)$ in the replicator dynamics and $f_{ij} = \mathcal{E}_{\mu_t}(p_ip_j)$ in replicator learning, we have

$$\begin{aligned}\frac{d}{dt}D(q|\bar{p}(t)) &= -\sum_i \frac{q_i}{\bar{p}_i(t)}\dot{\bar{p}}_i(t) = \{(5.81), (5.82)\} \\ &= -\sum_i \frac{q_i}{\bar{p}_i(t)} \sum_{j\neq i} \Delta_{ij}(t) f_{ij} \\ &= \sum_i \sum_{j>i} \Delta_{ij}(t)(q_j\bar{p}_i(t) - q_i\bar{p}_j(t)) \frac{f_{ij}}{\bar{p}_i(t)\bar{p}_j(t)}\end{aligned} \tag{5.90}$$

We now turn our attention to the sign of the expression $\Delta_{ij}(q_j\bar{p}_i(t) - q_i\bar{p}_j(t))$. With our parametrization of the ESS in the entries of the matrix, we have for $i, j \neq l$:

$$\begin{aligned}&\Delta_{ij}(t)(q_j\bar{p}_i(t) - q_i\bar{p}_j(t)) \\ &= (a_i\bar{p}_i(t) - a_j\bar{p}_j(t))(-\Pi_{m\neq j} a_m \bar{p}_i(t) + \Pi_{n\neq i} a_n \bar{p}_j(t)) \frac{1}{\Pi_{j\neq l} a_j - \sum_{m=1}^{l-1} \Pi_{j\neq m} a_j} \\ &= (a_i\bar{p}_i(t) - a_j\bar{p}_j(t))(-a_i\bar{p}_i(t) + a_j\bar{p}_j(t)) \frac{\Pi_{m\neq i,j} a_m}{\Pi_{j\neq l} a_j - \sum_{m=1}^{l-1} \Pi_{j\neq m} a_j} \\ &= -(a_i\bar{p}_i(t) - a_j\bar{p}_j(t))^2 \frac{\Pi_{m\neq i,j} a_m}{\Pi_{j\neq l} a_j - \sum_{m=1}^{l-1} \Pi_{j\neq m} a_j}\end{aligned}$$

If l is odd, $\Pi_{m\neq i,j} a_m > 0$ and $\Pi_{j\neq l} a_j - \sum_{m=1}^{l-1} \Pi_{j\neq m} a_j > 0$.

If l is even, $\Pi_{m\neq i,j} a_m < 0$ and $\Pi_{j\neq l} a_j - \sum_{m=1}^{l-1} \Pi_{j\neq m} a_j < 0$. (5.91)

If $j = l$, we have:

$$\begin{aligned}
&\Delta_{il}(t)(q_l\bar{p}_i(t) - q_i\bar{p}_l(t)) \\
&= (a_i\bar{p}_i(t) + a_l\bar{p}_l(t))(\Pi_{m\neq l}a_m\bar{p}_i(t) + \Pi_{n\neq i}a_n\bar{p}_l(t))\frac{1}{\Pi_{j\neq l}a_j - \sum_{m=1}^{l-1}\Pi_{j\neq m}a_j} \\
&= (a_i\bar{p}_i(t) + a_l\bar{p}_l(t))(a_i\bar{p}_i(t) + a_l\bar{p}_l(t))\frac{\Pi_{m\neq i,l}a_m}{\Pi_{j\neq l}a_j - \sum_{m=1}^{l-1}\Pi_{j\neq m}a_j} \\
&= (a_i\bar{p}_i(t) + a_l\bar{p}_l(t))^2\frac{\Pi_{m\neq i,l}a_m}{\Pi_{j\neq l}a_j - \sum_{m=1}^{l-1}\Pi_{j\neq m}a_j}
\end{aligned}$$

$$\begin{aligned}
&\text{If } l \text{ is odd, } \Pi_{m\neq i,l}a_m < 0 \text{ and } \Pi_{j\neq l}a_j - \sum_{m=1}^{l-1}\Pi_{j\neq m}a_j > 0. \\
&\text{If } l \text{ is even, } \Pi_{m\neq i,l}a_m > 0 \text{ and } \Pi_{j\neq l}a_j - \sum_{m=1}^{l-1}\Pi_{j\neq m}a_j < 0. \qquad (5.92)
\end{aligned}$$

Thus, $\Delta_{ij}(t)(q_j\bar{p}_i(t) - q_i\bar{p}_j(t)) < 0$ for all $i, j \in \{1, ..., l\}$, $i < j$.

Finally returning to equation (5.90), we see that with this choice of G, we indeed have for replicator learning that

$$\frac{d}{dt}D(q|\bar{p}(t)) = -\sum_i \frac{q_i}{\bar{p}_i(t)}\dot{\bar{p}}_i(t) \quad = \sum_i\sum_{j>i}\Delta_{ij}(t)(q_j\bar{p}_i(t) - q_i\bar{p}_j(t))\frac{f_{ij}}{\bar{p}_i(t)\bar{p}_j(t)} < 0 \qquad (5.93)$$

since $\forall i \neq j$, $i > j$, $\Delta_{ij}(t)(q_j\bar{p}_i(t) - q_i\bar{p}_j(t)) < 0$ and $\frac{f_{ij}}{\bar{p}_i(t)\bar{p}_j(t)} = \frac{\mathcal{E}(p_ip_j)}{\bar{p}_i(t)\bar{p}_j(t)} > 0$.

5.3.6 Convergence of Individual Dynamics

We can finally make the following statement concerning the expected movement of the individual units as $t \to \infty$:

Corollary 2 If $\lim_{t\to\infty}\bar{p}(t) = q$ where q is a totally mixed ESS, then the expected movement of each individual unit vanishes.

Proof By lemma 1, we have $E(e_i, q) = E(q, q)$ $\forall i = 1, ..., l$ since q is a totally mixed Nash equilibrium. Thus,

$$\begin{aligned}
&\lim_{t\to\infty}\int E(e_i - p, \mu_t(p))p_i d\mu_t \\
&= \lim_{t\to\infty}\int E(e_i - p, \bar{p}(t))p_i d\mu_t \\
&= \int E(e_i - p, q)p_i d\mu_t = 0 \qquad (5.94)
\end{aligned}$$

5.4 Control Theoretic Interpretation

Now returning to our original scenario, we will see how the individual strategies $p^k(t)$ can be interpreted as positions moving within the simplex.

5.4.1 Proposed Algorithm

We now use the notions presented above to describe our proposed approach in detail, starting by the case where there is only one waypoint y, where the m-ball B^m is the largest m-ball having its center at the centroid p^c of the simplex S_m delimited by the hyperplanes $p_i \geq 0 \ \forall i = 1, ..., m$, $\sum_{i=1}^{m} p_k \leq 1$ and B^m is contained within S_m - we will refer to this particular m-ball as B_0^m. We will then show how the algorithm for this particular case can be generalized to other m-balls and to sets of waypoints.

Idea In Brief

As we saw above, for any $l < \infty$ it is possible to construct an lxl game that has a totally mixed and thus unique ESS p. Furthermore, this ESS will be the global attractor in the replicator dynamics, as was also shown above. Thus, as long as the fraction p_i of any allowed phenotype i was initially nonzero, the composition of the population will approach p as $t \to \infty$.

If we see our units as a set of N populations growing acording to the replicator dynamics and assume that an individual from any population $k = 1, ..., N$ is equally likely to be matched against an individual from any of the other populations, we have N coupled units that update their compositions p^k in the same way.

Since $\forall k$, $\sum_{i=1}^{l} p_i^k = 1$, each p^k has $l-1$ degrees of freedom. Letting $l - 1 = m$, our approach consists in mapping the position $x^k(t)$ of each unit k to the composition $p^k(t)$ of population k and to map the waypoint y to the totally mixed ESS p of the game matrix defining the relative fitness. Our choice of B_0^m lets us interpret x^k both as a position in $B_0^m \subset \Re^m$ and as a probability distribution over an action space A of cardinality $l = m + 1$, as we will see below.

Position and Composition

Each position $x^k \in B_0^m \subset \Re^m$ can be mapped to a composition $p^k \in \Re^{m+1}$ by the following mapping $h(\cdot)$: $p_i^k = x_i^k$ if $i = 1, 2, ..., m$ and $p_{m+1}^k = 1 - \sum_{i=1}^{m} x_i^k$. We use the term composition instead of probability distribution to stress the analogy with the composition of a population.

Since $|\dot{p}(\cdot)| \leq C|G|$, for a positive constant $C < \infty$, by choosing G such that $|G|$ is small enough one can ensure that indeeed $|f(\cdot)| < v_{\max}$.

Local Information

Each unit knows only its own current position, not the position of any other unit or the waypoints. However, to estimate the average position of the other units, some information exchange with the other units is necessary - we will allow random interactions between units to occur according to the following protocol: at any time t, two units are selected at random and matched against each other in a game represented by its game matrix $G(t)$. $G(t)$ is constructed to have the composition $p = h(y)$ as its unique and totally mixed ESS - the reason for this choice will be given below. Each unit k selected uses its current composition $p^k(t) = h(x^k(t))$ to select an action of index i and observes the action of index j chosen by the other unit. If this is interpreted as a game, the resulting payoff can be looked up as entry $G(i, j)$ in the game matrix.

To describe this possible information exchange, we will introduce the random variable $I^k(t)$ to denote the information received by unit k at time t. If unit k is matched against another unit at time t, $I^k(t)$ will be set to j, the index of the action chosen by the opponent; otherwise, $I^k(t)$ will be given a null value 0 to indicate that no information was received. At any time t, the outcome of $I^k(t)$ depends on whether unit k is matched against another unit and on the current compositions of the other units. We will use the variable $i^k(t)$ to denote the outcome of $I^k(t)$ and, if $i^k(t) \neq 0$, denote by $\bar{p}^{i^k(t)}$ the pure strategy described by $\bar{p}_i^{i^k(t)} = 1$ if $i = i^k(t)$, $\bar{p}_i^{i^k(t)} = 0$ otherwise.

Local Algorithm

For each unit k, the algorithm is based on the following approximation of the replicator dynamics in discrete time:

$$p_i^k(t+1) = (1+E(e_i, \bar{p}^{i^k(t)}) - E(p^k(t), \bar{p}^{i^k(t)}) p_i^k(t) = (1+(e_i - p^k(t))^T G \bar{p}^{i^k(t)}) p_i^k(t)$$

or, equivalently for all pairs $\{(i, j) \mid i, j \in \{1, 2, ..., l\},\ i \neq j\}$:

$$\begin{aligned} \frac{p_i^k(t+1)}{p_j^k(t+1)} &= \frac{p_i^k(t)}{p_j^k(t)} + (E(e_i, \bar{p}^{i^k(t)}) - E(e_j, \bar{p}^{i^k(t)})) \frac{p_i^k(t)}{p_j^k(t)} \\ &= (1 + f_{ij}(i^k(t))) \frac{p_i^k(t)}{p_j^k(t)} \end{aligned} \tag{5.95}$$

When a unit is not matched against another unit, no update takes place, so we let $f_{ij}(0) = 0$. If unit k is indeed matched against another unit, we let $f_{ij}(I^k(t)) = G(i, I^k(t)) - G(j, I^k(t))$.

The algorithm is thus as follows:

- Get $i^k(t)$.
- Update $p^k(t+1)$
  so that for all pairs $\{(i,j) \mid i,j \in \{1,2,...,l\},\ i \neq j\}$:

  $\frac{p_i^k(t+1)}{p_j^k(t+1)} = (1 + f_{ij}(i^k(t)))\frac{p_i^k(t)}{p_j^k(t)}$ and $\sum_{i=1}^{l} p_i^k(t+1) = 1$.

- Update $x^k(t+1) = h^{-1}(p^k(t+1))$.

Assuming $i^k(t) = j$ and using the short notation G_{ij} for $G(i,j)$, this amounts to solving a system of linear equations to get $p^k(t+1)$ and applying $h^{-1}(\cdot)$ to obtain $x^k(t+1)$.
The system of linear equations is:

$$\begin{aligned}
\frac{p_1^k(t+1)}{p_2^k(t+1)} &= (1 + G_{1j} - G_{2j})\frac{p_1^k(t)}{p_2^k(t)} \\
\frac{p_2^k(t+1)}{p_3^k(t+1)} &= (1 + G_{2j} - G_{3j})\frac{p_2^k(t)}{p_3^k(t)} \\
&\vdots \\
\frac{p_{l-1}^k(t+1)}{1 - \sum_{i=1}^{l-1} p_i^k(t+1)} &= (1 + G_{l-1j} - G_{lj})\frac{p_{l-1}^k(t)}{p_l^k(t)}
\end{aligned} \tag{5.96}$$

Since $p^k(t) = h(x^k(t))$ and G_{ij} are known, this can be reformulated as $p^k(t+1) = A(t)p^k(t) + B(t)$ where $A(t)$ is a sparse matrix and $B(t)$ is a sparse vector.

From a local point of view, the algorithm aims at adapting the composition $p^k(t)$ to maximize the expected payoff. The average composition of the other units is estimated as the pure strategy corresponding to always playing the action of index j chosen by the current opponent. Thus, the problem of choosing a control to update one's current position is transformed into an estimation problem [54].

As noted above, if the initial values $p_i^k(0)$, $i = 1,...,l$, $k = 1,...,N$ satisfy the constraints $0 < p_i^k(0) < 1$ and $\sum_{i=1}^{l} p_i^k(0) = 1$, then the replicator dynamics assures that the same constraints are satisfied by the variables $p_i^k(t)$, $i = 1,...,l$, $k = 1,...,N$ for all $t > 0$. However, in a discrete time version implemented in a finite precision system, small errors may be introduced and one should therefore check that indeed $0 < p_i^k(t) < 1$ for all $t > 0$.

Generalization

How does the algorithm adapt to the case when $B^m \neq B_0^m$ and when there are more than just one waypoint y?

Different m-ball If $B^m \neq B_0^m$, we will choose the mapping $h(\cdot)$ such as to give $p_i^k(t) = \frac{R_m}{R} x_i^k(t) + p_i^c$ for $i = 1, ..., m$ and
$p_{m+1}^k = 1 - \sum_{i=1}^m p_i^k(t)$, $0 < p_i^k(t) < 1$ and $\sum_{i=1}^l p_i^k(t) = 1$, where R_m is a normalization constant and R is the radius of the ball B^m.

In the general case as in the special case presented above, by choosing G such that $|G|$ is small enough one can ensure that indeeed $|f(\cdot)| < v_{\max}$, since $|\dot{p}(\cdot)| \leq C|G|$, for a positive constant $C < \infty$.

Sets of Waypoints If we are dealing with a set of waypoints $y_1, y_2, ..., y_M$ rather than with just one single waypoint y, the modification needed will be to use a different matrix G_i for each waypoint y_i and to switch game matrices as one waypoint has been reached and it is time to move to the next one.

5.4.2 High-Level Control

From the perspective of the individual node, the control applied is thus a function of an estimate of the current average strategy [54]; as seen above, the crude estimate used is the pure strategy - or direction of motion - corresponding to the action chosen by the opponent in the game played.

This individual control assures that the collective goal of reaching the waypoint will be achieved; the algorithm can therefore be seen as a high-level control that leaves open the option of adding further individual control as seen in the heterogeneous case, further described below, to accommodate individual preferences in trajectories or individual end points. Thus, individual nodes still are at some liberty to choose components of the function $f(\cdot)$ introduced in the problem statement at the beginning of the chapter.

The essential part of this high-level control is thus in choosing a matrix G that has as its unique ESS the equilibrium corresponding to the desired waypoint; for any $l < \infty$ and any interior ESS, such a matrix G can easily be constructed by solving an underconstrained system of linear equations. Through the choice of G, a collective reference trajectory is thus generated.

5.4.3 What Is Learned?

In the introduction to the chapter, we briefly commented on the name replicator learning and what was actually learned by the individual units and by the system as a whole, respectively. From a game theoretic perspective, it is again the game theoretic equilibrium that is learned by the collective as a whole since the average straegy will correspond to the ESS. As noted above, from a motion control point of view, the units learn how to mutually adapt their positions relative to each other so as to make the average position coincide with the desired waypoint.

Remembering that replicator learning could be seen as a generalization of the two-population version of the multi-population replicator dynamics, we get a clearer picture of how this adaptation works. In the two-population replicator dynamics, the only attractors are asymmetric Nash equilibria in which one population chooses an aggressive strategy that maximizes its payoff and the other population opts for a defensive strategy to minimize its losses.

In our scenario, the players will typically encounter different opponents each time the game is played and take a step towards one of the asymmetric Nash equilibria, that is, towards becoming an aggressive or defensive player, depending on the matching.

This mutual adaptation may be seen as a self-organization of the units so that their average position coincides with a desired point; we note that with the replicator dynamics, where the individual units are static, all individual units would be situated permanently at the corners of the strategy simplex whereas in our case, they move within the simplex until the desired collective structure has been reached.

5.4.4 Relevant Applications

The most direct application of this scenario is a collective search for a lost object, where an estimate is available of the position of the object. Assuming each unit can observe points within a ball of a certain radius r_{obs} around it, one wants these observation fields to cover as large an area as possible and therefore wants to spread out the units as much as possible over the area where the lost object is most likely to be found.

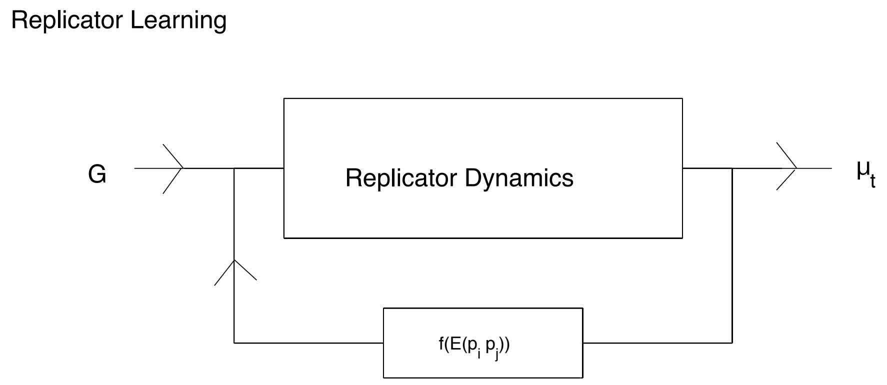

Figure 5.4: High-level control interpretation of the replicator learning.

5.5 Heterogeneity

If some units perform optimally within a particular interval in the node space, can we modify the given framework to ensure that not only does the average position converge to the current waypoint, but that in addition, those units associated with optimal intervals converge towards points in the respective intervals, provided that a feasible solution exists? Below, we present a modification of the algorithm that achieves this.

Two Objectives The added constraint of making sure that the heterogeneous nodes converge to their optimal points results in the system now having two simultaneous objectives, namely

- The average position should converge to the current waypoint
- The individual positions of the heterogeneous nodes should converge to points within their optimal operating intervals.

We propose to let the heterogeneous nodes themselves assure their convergence to their optimal intervals whereas the other nodes take care of the first and original objective, the convergence of the average position to the current waypoint. This is done by keeping the same interface as before but using different matrices for homogeneous and heterogeneous nodes.

Same Interface, Different Matrices We propose to use the same interface as before between units but add an alternative set of game matrices $\mathbf{H}^k$ that each has the preferred point of unit k as its unique equilibrium. If a set of points is preferred rather than just one point, the point set will be assumed to be a simplex and $\mathbf{H}^k$ will be the weighted sum of the matrices $\mathbf{H}^{k_1}, \mathbf{H}^{k_2}, ...$, which have the simplex vertices as their respective unique equilibria.

Heterogeneous Nodes Each heterogeneous node k will use the matrix $\mathbf{H}^k$ instead of the current game matrix $\mathbf{G}$ and will ignore the action chosen by its opponent in each matching; instead, it will use its own current composition $\mathbf{p}^k(t)$ to generate a ficticious action which will replace the action chosen by the opponent in the algorithm. Why is this? In fact, in this way, node k will simulate the replicator dynamics driven by the game matrix $\mathbf{H}^k$, which, as shown above, is proven to make the composition - in this case, $\mathbf{p}^k$ - converge to the ESS of $\mathbf{H}^k$, in this case, the optimal operating point.

In the case where a set of points is preferred rather than just one point, node k will simulate a weighted sum of replicator dynamics and will converge to a weighted average of the ESS's of the matrices used.

With this approach, unit k will thus disregard the collective goal of reaching the waypoint and focus entirely on attaining a point within its optimal interval. However, by making unit k take turns using the common game matrix $\mathbf{G}$ and its own particular matrix $\mathbf{H}^k$, the unit can both actively make the average position move towards the current waypoint and at the same time move towards its optimal operating interval.

Homogeneous Nodes Each homogeneous node will use the original algorithm without any modification, always using the game matrix $\mathbf{G}$ and always taking the action chosen by its opponent into account. Since the heterogeneous nodes will converge towards points in their respective optimal operating intervals, the homogeneous nodes will get an estimate of the average position that is different from the case where all nodes are homogeneous, since the heterogeneous nodes will not actively contribute to the convergence of the average position to the current waypoint - they are using their own game matrices $\mathbf{H}^k$. However, the heterogeneous units will contribute passively to the convergence of the average position by being taken into account and compensated for by the homogeneous nodes.

Example An example will illustrate the proposed modification to the given framework: in a system of $N = 100$ nodes moving on the unit circle, let us assume that the current waypoint is $y_1^2 = (1, \pi)$ in polar coordinates but that nodes $1-30$ operate optimally at waypoint $y_4^2 = (1, \frac{7\pi}{10})$. The other nodes are assumed to operate equally well at any point.

Units $1 - 30$ will use the matrix $\mathbf{H}$ which has as its unique ESS the composition $\mathbf{p}_H = (\frac{7}{20}, \frac{13}{20})$; the homogeneous nodes will use the game matrix $\mathbf{G}_1^2$, with $\mathbf{p}_1^2 = (\frac{1}{2}, \frac{1}{2})$ as its unique ESS.
The corresponding waypoints are

$$y_H = h(\mathbf{p}_H) = (1, 2\pi p_{H_1}) = (1, \frac{7\pi}{10}) \text{ and}$$
$$y_1^2 = h(\mathbf{p}_1^2) = (1, 2\pi p_{1_1}^2) = (1, \pi)$$

The simulation of this example can be seen to the left in figures (5.13) and (5.14) below; matrices $\mathbf{G}_1^2$ and $\mathbf{H}$ are given directly below.

$$\mathbf{G_1^2} = \begin{array}{ccc} Player1/Player2 & Action1 & Action2 \\ Action1 & (-1,-1) & (2,0) \\ Action2 & (0,2) & (1,1) \end{array}$$

$$\mathbf{H} = \begin{array}{ccc} Player1/Player2 & Action1 & Action2 \\ Action1 & (-13,-13) & (14,0) \\ Action2 & (0,14) & (7,7) \end{array}$$

5.6 Reconfiguration

How are the three classes of reconfigurations introduced above interpreted in this context?

5.6.1 System Reconfiguration

System reconfigurations are changes at the global level that take place without any loss or addition of the constituent parts of the network, that is, the nodes. Since the global property that we are interested in here is the average node position, a system reconfiguration in this context is the shift of the average position from one waypoint to another one.

Below, we will see simulations of all three classes of reconfigurations which all prove to be stable and have rapid convergence. Why do the reconfigurations work? At each reconfiguration initiation, the time can be seen as reset to zero and the system starts out with a new game matrix G; as the time goes to infinity, the average strategy converges to the unique ESS of G, that is, the average position converges to the current waypoint.

5.6.2 Permutation Reconfiguration

Permutation reconfigurations are changes at the individual level taking place while at the global level, the system keeps the same configuration. In this application, our main interest is the average properties of the network - few assumptions are made about the particular positions occupied by the individual nodes. However, as seen above, our framework can be adapted to accommodate heterogeneous nodes performing optimally at particular positions - the adaptation consisted in using special game matrices for the heterogeneous nodes while keeping the same interface as before.

Since in this model of heterogeneity, the individual positions of some nodes are actually taken into account, a permutation reconfiguration can be achieved by letting heterogeneous nodes switch preferences or by making nodes drop or adopt preferences.

5.6.3 Structure Reconfiguration

Structure reconfigurations are changes at the global level accompanied by a change of the actual network constituents such as the loss or addition of nodes. As opposed to a formation control context, where the position of each node is clearly defined relative to all other nodes, in this setting a structure reconfiguration does not in general change the configuration space - that would only be the case if all existing nodes were heterogeneous ones with restricted action spaces.

Still, an adddition or subtraction of nodes will cause a temporary disturbance of the average position from the current waypoint, as we will see in the simulations below. This disturbance will automatically be discarded by the system dynamics, either by the new nodes adapting their strategies to the game played or by the other nodes shifting their strategies to compensate for the disturbance.

Biological analogies to structure reconfigurations are massive cell losses in the human liver or brain; the liver can still function after up to 80% of its cells are lost and the brain can to a variable extent redistribute functions such as speech control to other parts of the brain after local massive cell loss.

5.6.4 Reconfiguration Initiation

The given framework offers many ways, from hierarchical to purely distributed, for initiating reconfigurations. Below are the most important ones, which can be used separately or combined.

Broadcast Signal - Push the Button Any node can broadcast a signal for reconfiguration with the index of the desired waypoint or even with the game matrix. This can be thought of as any node pushing a button; this must however be combined with a period of inactivation after each command to avoid conflicting orders. Such orders can also be issued by a centralized coordinator, in which case no inactivation period is needed.

Synchronized Switch All units can have synchronized clocks and agree in advance on successive points in time at which reconfigurations will occur and on a predefined reconfiguration order; natural rhythms such as change of tides, light, temperature and magnetic field strength are some of the signals that may serve as clocks.

Measuring Others' Average Difference Before the current waypoint is reached, the average of the estimate of the opponent action over time will change, whereas it will be constant after the waypoint is reached. By keeping track of the actions chosen by its opponents, each node can thus get information about whether the waypoint is reached. If the average node position was already close to the waypoint at the start, the average will be almost constant from the start.

Measuring Own Average Difference The node can also measure the average of its own position over time - before the waypoint is reached, it will change on an average whereas this will no longer be the case after the waypoint is reached.

5.7 Simulation Results

To illustrate our theory, we studied the dynamics of a system of $N \geq 100$ units moving on the unit circle or in $\mathbf{S}_2 \subset \Re^2$ or $\mathbf{S}_3 \subset \Re^3$, respectively, and updating their current positions according to the proposed algorithm. A number of different waypoints and initial conditions were used to test the algorithm and are given in the first section below.

We first studied the convergence to one waypoint at a time and then simulated successive system reconfigurations from waypoint to waypoint. Structure reconfigurations were then simulated in which node losses or additions of up to 50% were studied. Finally, we showed non-convergence when the game matrix used did not have an ESS.

5.7.1 Waypoints and Initial Conditions

To test our algorithm in simulations, we used different waypoints and three main types of initial conditions, all presented below.

Waypoints on the Unit Circle To study our model of a walk on the unit circle, we chose the following four different waypoints y_i^2 in polar coordinates:

$$y_1^2 = (1, \pi),\ y_2^2 = (1, \tfrac{\pi}{2}),\ y_3^2 = (1, \tfrac{7\pi}{4}) \text{ and } y_4^2 = (1, \tfrac{7\pi}{10}).$$

The corresponding compositions $\mathbf{p}_i^2$ are obtained as

$$\mathbf{p}_i^2 = h^{-1}(y_i^2) = (\tfrac{1}{2\pi} y_i^2, 1 - \tfrac{1}{2\pi} y_i^2); \text{ they are}$$

$$\mathbf{p}_1^2 = (\tfrac{1}{2}, \tfrac{1}{2}),\ \mathbf{p}_2^2 = (\tfrac{1}{4}, \tfrac{3}{4}),\ \mathbf{p}_3^2 = (\tfrac{7}{8}, \tfrac{1}{8}) \text{ and } \mathbf{p}_4^2 = (\tfrac{7}{20}, \tfrac{13}{20}).$$

Four game matrices $\mathbf{G}_i^2$ were chosen to have the corresponding compositions $\mathbf{p}_i^2$ as unique ESS's; the game matrices and their respective ESS's are given below.

$$\mathbf{G_1^2} = \begin{array}{ccc} Player1/Player2 & Action1 & Action2 \\ Action1 & (-1,-1) & (2,0) \\ Action2 & (0,2) & (1,1) \end{array}$$

$$\mathbf{p}_1^2 = (\tfrac{1}{2}, \tfrac{1}{2})$$

$$\mathbf{G_2^2} = \begin{array}{ccc} Player1/Player2 & Action1 & Action2 \\ Action1 & (-3,-3) & (2,0) \\ Action2 & (0,2) & (1,1) \end{array}$$

$$\mathbf{p}_2^2 = (\tfrac{1}{4}, \tfrac{3}{4})$$

$$\mathbf{G_3^2} = \begin{array}{ccc} Player1/Player2 & Action1 & Action2 \\ Action1 & (-1,-1) & (14,0) \\ Action2 & (0,14) & (7,7) \end{array}$$

$$\mathbf{p}_3^2 = (\tfrac{7}{8}, \tfrac{1}{8})$$

$$\mathbf{G_4^2} = \begin{array}{ccc} Player1/Player2 & Action1 & Action2 \\ Action1 & (-13,-13) & (14,0) \\ Action2 & (0,14) & (7,7) \end{array}$$

$$\mathbf{p}_4^2 = (\tfrac{7}{20}, \tfrac{13}{20})$$

Waypoints in $\mathbf{S}_2 \subset \Re^2$ To study the motion of the nodes in $\mathbf{S}_2 \subset \Re^2$, we selected the following four waypoints $\mathbf{y}_i^3$:

$$\mathbf{y}_1^3 = (0.6, 0.2),\ \mathbf{y}_2^3 = (0.1, 0.4),$$
$$\mathbf{y}_1^3 = (0.33, 0.33) \text{ and } \mathbf{y}_4^3 = (0.05, 0.15).$$

Four game matrices $\mathbf{G}_i^3$ were chosen to have the corresponding compositions $\mathbf{p}_i^3 = h^{-1}(\mathbf{y}_i^3) = (y_{i1}^3, y_{i2}^3, i - y_{i1} - y_{i2})$ as unique ESS's; the game matrices and their respective ESS's are given below.

$$\mathbf{G_1^3} = \begin{array}{cccc} Player1/Player2 & Action1 & Action2 & Action3 \\ Action1 & (6,6) & (5,7) & (5,7) \\ Action2 & (7,5) & (4,4) & (3,5) \\ Action3 & (7,5) & (5,3) & (2,2) \end{array}$$

$$\mathbf{p}_1^3 = (\tfrac{3}{5}, \tfrac{1}{5}, \tfrac{1}{5})$$

$$\mathbf{G_2^3} = \begin{array}{cccc} Player1/Player2 & Action1 & Action2 & Action3 \\ Action1 & (12,12) & (2,26) & (4,3) \\ Action2 & (26,2) & (1,1) & (2,8) \\ Action3 & (3,4) & (8,2) & (1,1) \end{array}$$

$$\mathbf{p}_2^3 = (\tfrac{1}{10}, \tfrac{4}{10}, \tfrac{5}{10})$$

$$\mathbf{G_3^3} = \begin{array}{cccc} Player1/Player2 & Action1 & Action2 & Action3 \\ Action1 & (1,1) & (5,8) & (6,6) \\ Action2 & (8,5) & (2,2) & (2,5) \\ Action3 & (6,6) & (5,2) & (1,1) \end{array}$$

$$\mathbf{p}_3^3 = (\tfrac{1}{3}, \tfrac{1}{3}, \tfrac{1}{3})$$

$$\mathbf{G_4^3} = \begin{array}{cccc} Player1/Player2 & Action1 & Action2 & Action3 \\ Action1 & (2,2) & (2,5) & (2,6) \\ Action2 & (5,2) & (1,1) & (2,6) \\ Action3 & (6,2) & (6,2) & (1,1) \end{array}$$

$$\mathbf{p}_4^3 = (\tfrac{1}{20}, \tfrac{3}{20}, \tfrac{16}{20})$$

Waypoints in $\mathbf{S}_3 \subset \Re^3$ To simulate three-dimensional motion in $\mathbf{S}_3 \subset \Re^3$, we finally chose the following two waypoints $\mathbf{y}_i^4$:

$$\mathbf{y}_1^4 = (0.3, 0.4, 0.2) \text{ and } \mathbf{y}_2^4 = (0.25, 0.25, 0.25).$$

The two matrices $\mathbf{G}_i^4$ chosen to have the corresponding compositions $\mathbf{p}_i^4 = h^{-1}(\mathbf{y}_i^4) = (y_{i1}^4, y_{i2}^4, y_{i3}^4, 1 - y_{i1}^4 - y_{i2}^4 - y_{i3}^4)$ as ESS's are given below along with their ESS's.

$$\mathbf{G_1^4} = \begin{array}{ccccc} Player1/Player2 & Action1 & Action2 & Action3 & Action4 \\ Action1 & (2,2) & (15,4) & (10,16) & (14,8) \\ Action2 & (4,15) & (5,5) & (20,10) & (28,5) \\ Action3 & (16,10) & (10,20) & (2,2) & (8,25) \\ Action4 & (8,14) & (5,28) & (25,8) & (6,6) \end{array}$$

$$\mathbf{p}_1^4 = (\tfrac{3}{10}, \tfrac{4}{10}, \tfrac{2}{10}, \tfrac{1}{10})$$

$$\mathbf{G_2^4} = \begin{array}{ccccc} Player1/Player2 & Action1 & Action2 & Action3 & Action4 \\ Action1 & (1,1) & (4,8) & (5,4) & (10,3) \\ Action2 & (8,4) & (2,2) & (6,12) & (4,7) \\ Action3 & (4,5) & (12,6) & (2,2) & (2,9) \\ Action4 & (3,10) & (7,4) & (9,2) & (1,1) \end{array}$$

$$\mathbf{p}_2^4 = (\tfrac{1}{4}, \tfrac{1}{4}, \tfrac{1}{4}, \tfrac{1}{4})$$

Initial Conditions We used three main types of qualitatively different initial conditions to test our algorithm. Firstly, we used the identical initial condition to let all nodes start at the same position, thus obtaining a coherent swarm provided that all nodes took small enough steps.

Secondly, we let the units start from points chosen uniformly at random, which gave a system particularly robust to node losses of even more than 50%.

Finally, we distributed the nodes as far from the centroid of $\mathbf{S}_2$ or $\mathbf{S}_3$ as possible, thus obtaining an initial node distribution that could be described an anti-swarm.

In the section on structure reconfigurations, we used a fourth qualitatively different type of initial condition in order to obtain a noticeable disturbance when up to 50% of the nodes are lost - this is the biidentical initial distribution, in which half of the nodes start at one position and the other half at another position.

5.7.2 Convergence to Unique ESS

We first studied the convergence of the average position to the respective waypoints using different combinations of initial positions, waypoints and number of units N - in all cases, the average position was found to converge to the desired waypoint. Our simulation results are presented below, showing the dynamics of the average position and the covariances.

Waypoints on the Unit Circle Using the mapping $y^2 = 2\pi p_1^2$ introduced above, where p_1^2 is the first component of $\mathbf{p}^2$, we obtain the waypoints as positions $(1, \theta)$ on the unit circle. To the left in figures (5.5) and (5.6), we see convergence of the average position to the waypoints y_1^2 and y_3^2, respectively; in both cases, all units started from the same point. To the right in the figures, we see the final positions of the individual units.

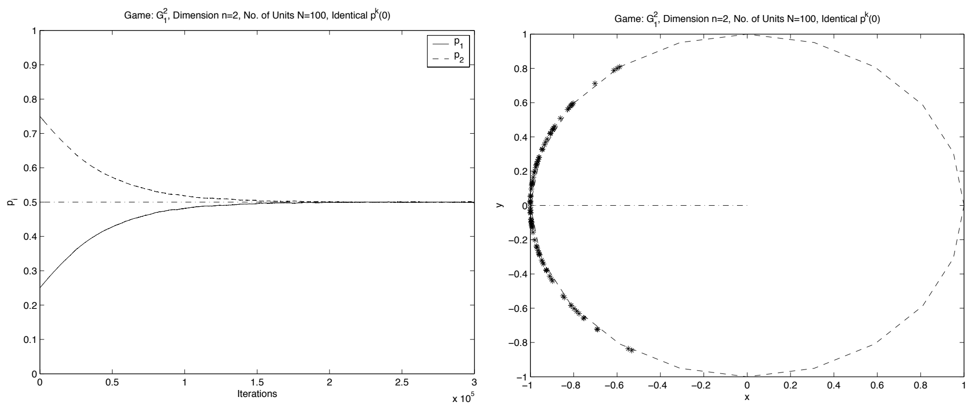

Figure 5.5: Convergence to y_1^2 to the left, corresponding final individual positions to the right; all units starting from the same point, N=100.

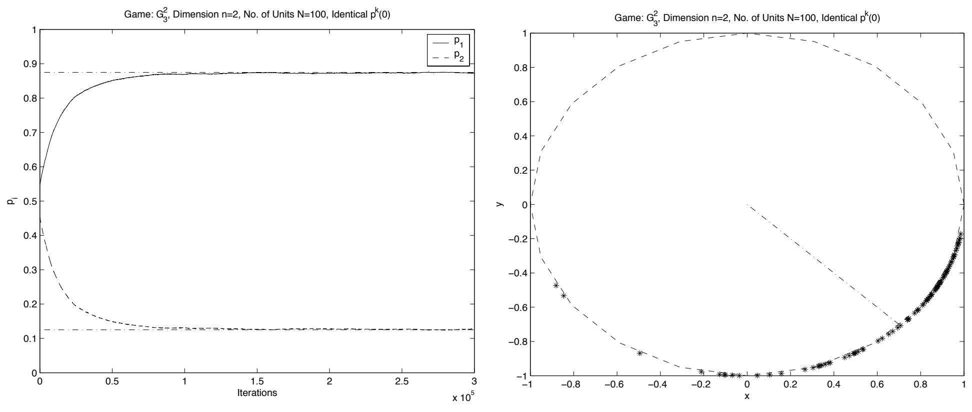

Figure 5.6: Convergence to y_3^2 to the left, corresponding final individual positions to the right; all units starting from the same point, N=100.

Waypoints in $\mathbf{S}_2 \subset \Re^2$ With the mapping $\mathbf{y}^3 = (p_1^3, p_2^3)$, we obtain the waypoints as positions in $\mathbf{S}^2$, introduced above; as noted above, this mapping can be generalized by scaling and translation to $\mathbf{y}^3 = \alpha(p_1^3, p_2^3) - (\beta_1, \beta_2)$.

To the left in figures (5.7), (5.8), (5.9) and (5.10), we see the convergence of the average position to the waypoints $\mathbf{y}_1^3$, $\mathbf{y}_2^3$, $\mathbf{y}_3^3$ and $\mathbf{y}_4^3$, respectively; to the right in the figures, we see the respective covariances as functions of time. In figure (5.7), the units started from points far from the centroid of $\mathbf{S}_2$, in figures (5.8) and (5.10), the units started from points chosen uniformly at random over $\mathbf{S}_2$ whereas in figure (5.9), all units started from the same point.

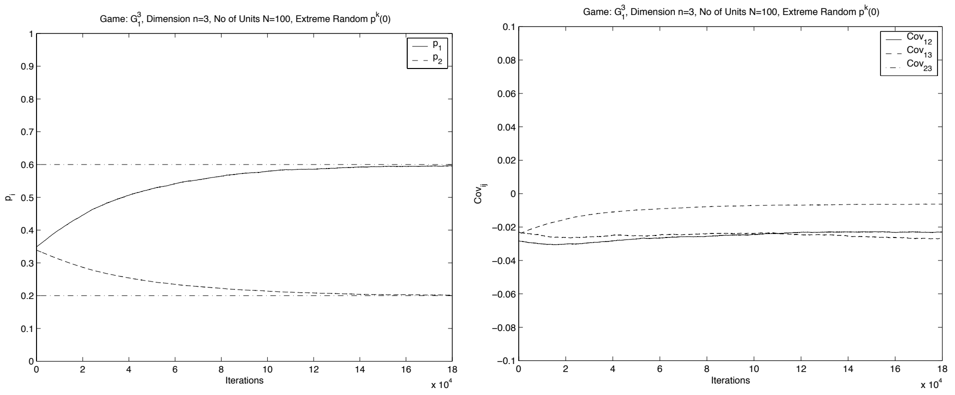

Figure 5.7: Convergence to $\mathbf{y}_1^3$ to the left, corresponding covariances to the right; units starting from points far from centroid, N=100.

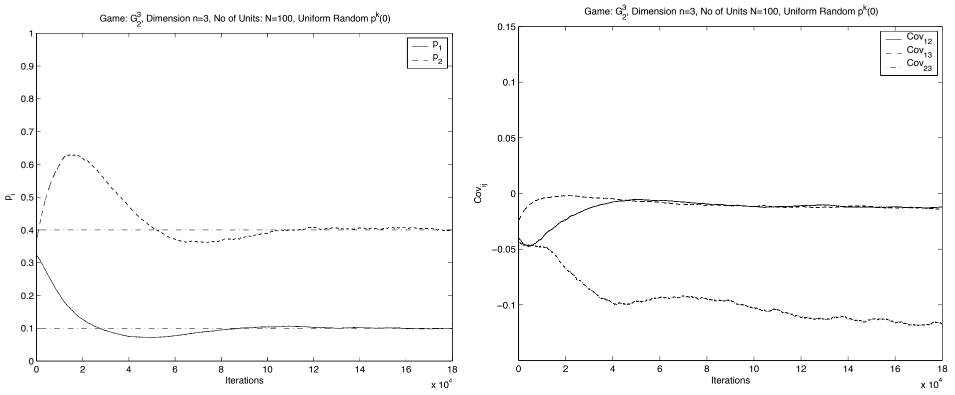

Figure 5.8: Convergence to $\mathbf{y}_2^3$ to the left, corresponding covariances to the right; units starting from points uniformly chosen at random, N=100.

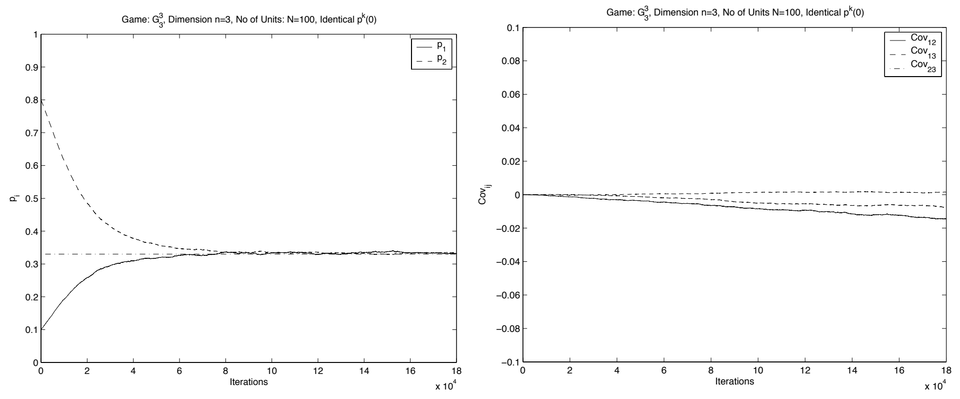

Figure 5.9: Convergence to $\mathbf{y}_3^3$ to the left, corresponding covariances to the right; all units starting from the same point, N=100.

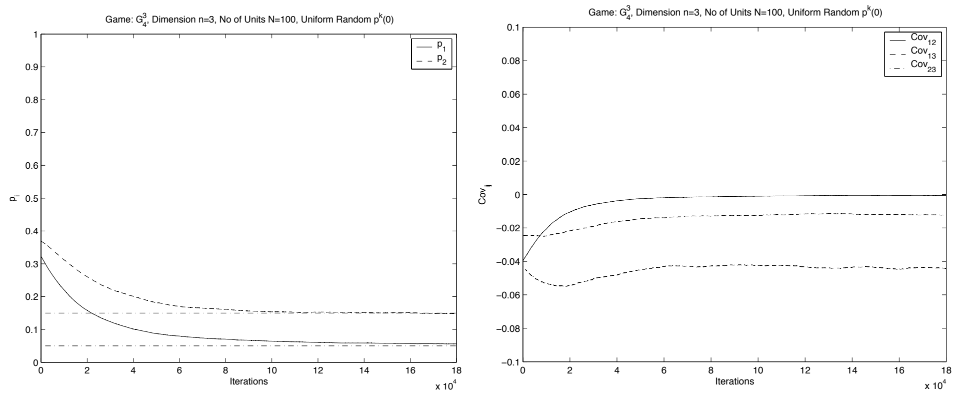

Figure 5.10: Convergence to $\mathbf{y}_4^3$ to the left, corresponding covariances to the right; units starting from points uniformly chosen at random, N=100.

Waypoints in $\mathbf{S}_3 \subset \Re^3$ Using the mapping $\mathbf{y}^4 = (p_1^4, p_2^4, p_3^4)$, finally, we obtain waypoints in $\mathbf{S}_3$, introduced above; as noted above, the mapping can be generalized by scaling and translation to
$\mathbf{y}^4 = \alpha(p_1^4, p_2^4, p_3^4) - (\beta_1, \beta_2, \beta_3)$.

To the left in figures (5.11) and (5.12), we see how the average position converges to the waypoints $\mathbf{y}_1^4$ and $\mathbf{y}_2^4$, respectively; to the right in the figures, we see the covariances as functions of time.

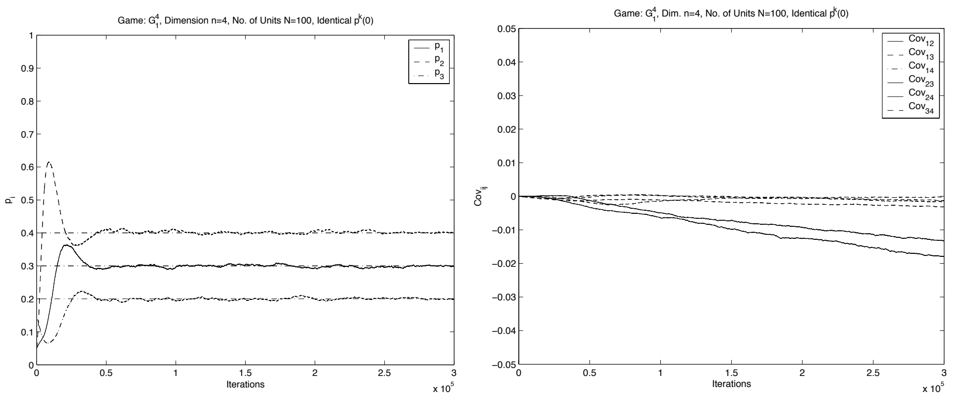

Figure 5.11: Convergence to $\mathbf{y}_1^4$ to the left, corresponding covariances to the right; units starting from points far from centroid, N=100.

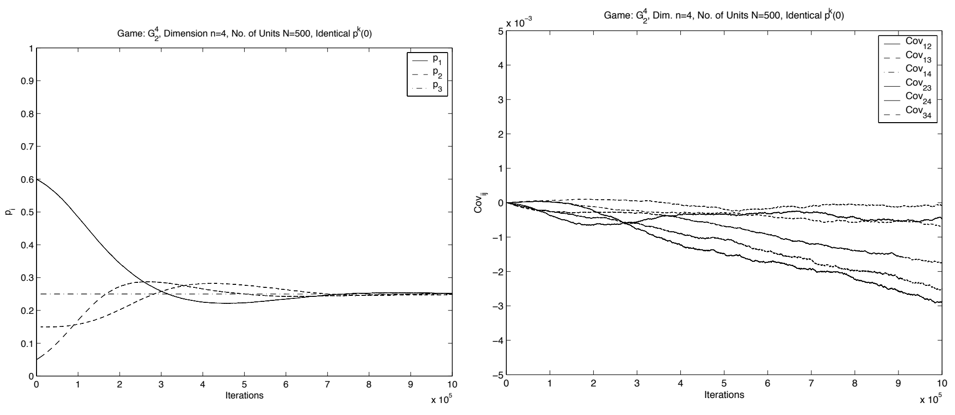

Figure 5.12: Convergence to $\mathbf{y}_2^4$ to the left, corresponding covariances to the right; units starting from points uniformly chosen at random, N=100.

5.7.3 Heterogeneity

The heterogeneous case described above will now be illustrated in simulation. Assuming that in a network of $N = 100$ nodes, thirty nodes are heterogeneous, operating optimally at a particular point y_{opt}, we let the system converge to the waypoint y_1^2 while simultaneously accommodating the preferences of the heterogeneous nodes. The point y_{opt} was chosen to be first y_4^2, then y_3^2.

To the left in figure (5.13) below, we see the convergence of the average position to the waypoint y_1^2 in the presence of thirty heterogeneous nodes; it is very similar to the homogeneous case shown above in figure (5.5). The final individual positions of all nodes were plotted for the homogeneous case to the right in figure (5.13) and for the two heterogeneous cases in figure (5.14).

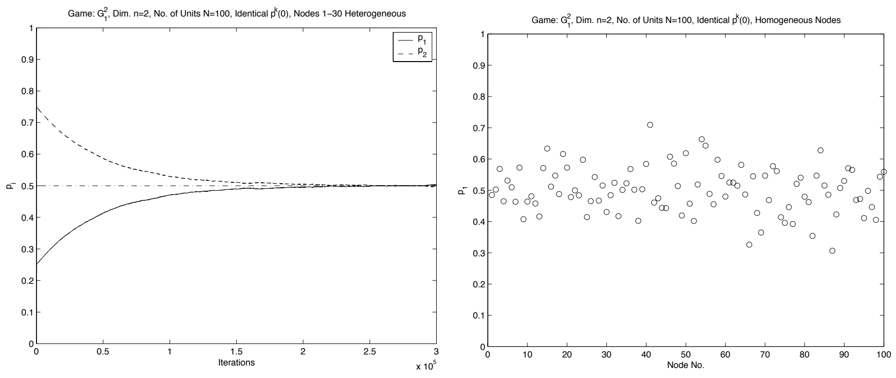

Figure 5.13: Convergence to y_1^2 in the presence of heterogeneous nodes, to the left; compare with the homogeneous case in figure (5.5). Final individual positions in the homogeneous case, to the right.

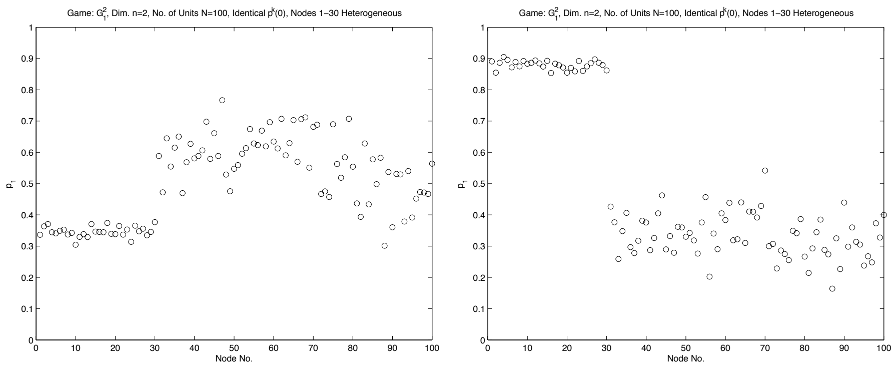

Figure 5.14: Final individual positions in the heterogeneous case; preference for y_4^2, to the left, preference for y_3^2, to the right.

5.7.4 Convergence to Either of Two ESS's

In a game that has two ESS's, both ESS's are attractors and the convergence to either of them will be a function of the initial strategies. Below, we see this illustrated in two different simulations for the 2×2 matrix G_5^2 which has as ESS's the strategies $\mathbf{p}_{5a}^2 = (1,0)$ and $\mathbf{p}_{5b}^2 = (0,1)$.

We chose the initial strategies of the individual players at random according to a uniform probability distribution and got convergence to $\mathbf{y}_{5a}^2$ as seen to the left in figure (5.15). The corresponding initial strategies are shown to the left in figure (5.16).

$$\mathbf{G_5^2} = \begin{array}{ccc} Player1/Player2 & Action1 & Action2 \\ Action1 & (1,1) & (-1,-1) \\ Action2 & (-1,-1) & (1,1) \end{array}$$

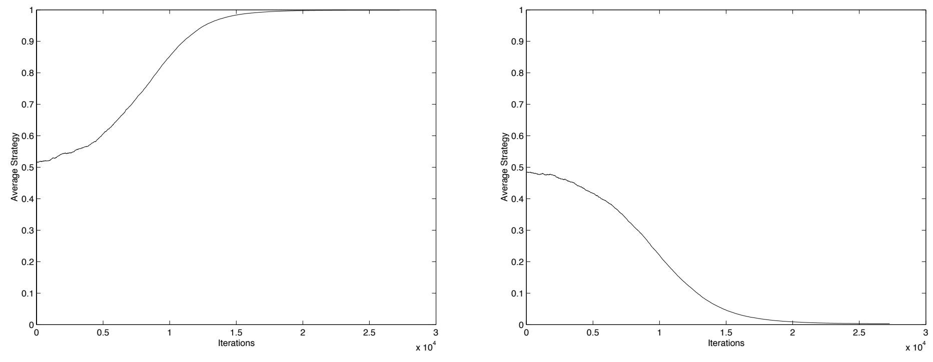

Figure 5.15: Convergence to waypoints (1, 2π) and (1, 0) for G_5^2, $N = 100$.

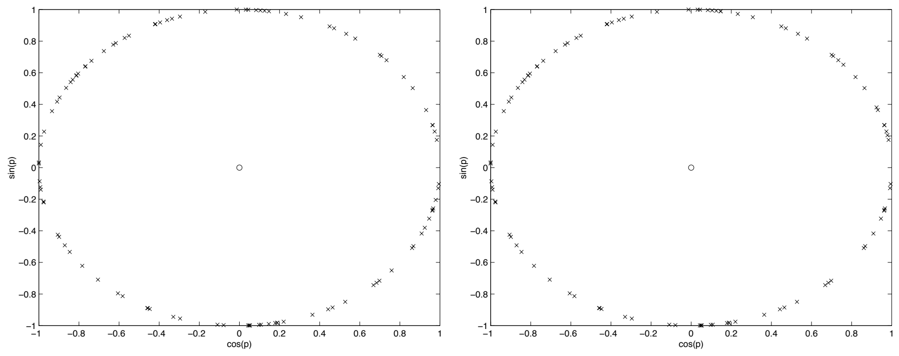

Figure 5.16: Individual initial strategies for G_5^2 giving convergence to (1, 2π) to the left and to (1, 0) to the right, $N = 100$.

Sensitivity We then picked four players whose initial strategies $p^k = (p_1^k, 1 - p_1^k)$ were close to $\mathbf{p}_{5a}^2$ and reversed their initial strategies to $\hat{p}^k =$

$(1-p_1^k, p_1^k)$ while letting the other ninety-six players keep their initial strategies. The system now converged to $\mathbf{y}_{5b}^2$ instead, as seen to the right in figure (5.15), with the corresponding initial strategies shown to the right of figure (5.16).

Equivalent Waypoints Thus, using game matrices that have more than one ESS will give a capricious system that converges to either of the ESS's depending on the initial conditions. However, if the two ESS's are seen as equivalent, we can see this as analogous to the unique ESS case discussed above. Indeed, if we continue using the mapping introduced above, where the waypoints were positions on the unit circle, both the above ESS's would correspond to the position $(1,0)$ on the unit circle. Thus, although toggling the coordinates of 4% of the units would lead the system to approach the waypoint $(1,0)$ clockwise rather than counter-clockwise, the system would still converge to the same waypoint on the unit circle.

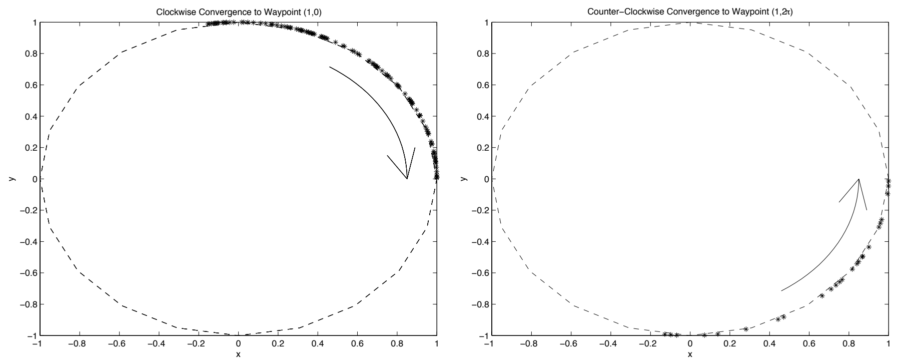

Figure 5.17: Waypoints $(1,0)$ and $(1,2\pi)$ in polar coordinates correspond to the same point approached clockwise or counter-clockwise.

$$\mathbf{G_{HD}} = \begin{array}{ccc} Player1/Player2 & Action1 & Action2 \\ Action1 & (-1,-1) & (2,0) \\ Action2 & (0,2) & (1,1) \end{array}$$

Reconfiguration By Information Remembering now the discussion from the section on game theory on how the addition of information may change the equilibrium set of a game and the particular example of the game of Hawks and Doves $\mathbf{G_{HD}}$ with an asymmetry, this represents an alternative way of switching games: the game is different not because the game parameters were changed, but because more or less information was given to the players.

As we remember, when no information is provided as to which player is cast in which role $i \in I = \{1,2\}$, the game has one unique ESS

$\mathbf{p^{HD}} = (0.5, 0.5)$, corresponding to the waypoint $(1, \pi)$ in polar coordinates. However, when information about the asymmetry is provided and taken into account by all players, the resulting game has two ESS's $\mathbf{p_1^{HD_{asymm}}}(j|i)$ and $\mathbf{p_2^{HD_{asymm}}}(j|i)$, both given below and conditional on the information obtained about the role i.

$$\mathbf{p_1^{HD_{asymm}}}(j|i) = 1 \text{ if } i = j \text{ and } \mathbf{p_1^{HD_{asymm}}}(j|i) = 0 \text{ if } i \neq j \text{ whereas}$$
$$\mathbf{p_2^{HD_{asymm}}}(j|i) = 0 \text{ if } i = j \text{ and } \mathbf{p_2^{HD_{asymm}}}(j|i) = 1 \text{ if } i \neq j.$$

Again using the same mapping to obtain waypoints on the unit circle, $\mathbf{p_1^{HD_{asymm}}}(j|i=0)$ and $\mathbf{p_2^{HD_{asymm}}}(j|i=1)$ correspond to the waypoint $(1, 0)$ in polar coordinates whereas $\mathbf{p_2^{HD_{asymm}}}(j|i=0)$ and $\mathbf{p_1^{HD_{asymm}}}(j|i=1)$ give the waypoint $(1, 2\pi)$ - because of the periodicity with period 2π, the two waypoints actually correspond to the same point.

With a slight modification of the above mapping, we can now make the system reconfigure by giving or withholding information about the asymmetry - the modification consists in consistently using either $\mathbf{p^{HD_{asymm}}}(j|i=0)$ or $\mathbf{p^{HD_{asymm}}}(j|i=1)$ to obtain the waypoint when information is provided.

Simulations of reconfigurations caused by a change of game parameters will be shown next; below, we see reconfigurations caused by the addition or reduction of information, respectively.

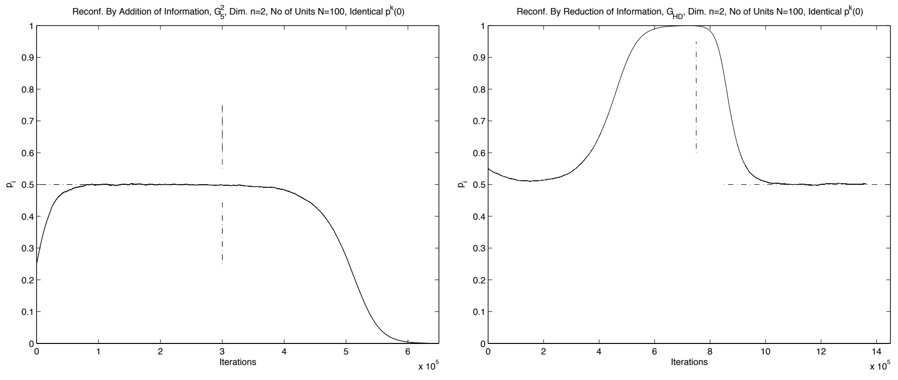

Figure 5.18: Reconfiguration by addition of information, to the left, and by reduction of information, to the right.

5.7.5 Reconfiguration

We also studied the reconfiguration dynamics of our system, making our system of $N = 100$ units reconfigure in $\Re$ and $\Re^2$ - all three categories of reconfigurations were represented and the results are shown below, starting with system reconfigurations, continuing with structure reconfigurations and ending with permutation reconfigurations.

System Reconfiguration In a system reconfiguration, the global network structure changes while the network components remain the same. In our case, the relevant system component is the average node position, which is a function of the current game matrix G - we simulated a reconfiguration by imposing a discontinuous switch of game matrices. Figures (5.19) and (5.20) show successive reconfigurations in $\Re$. In figure (5.19), the system reconfigures from $y_1^2 \to y_2^2 \to y_3^2$, whereas in figure (5.20), we see a reconfiguration from $y_3^2 \to y_1^2 \to y_4^2$; the waypoints are shown as dotted reference lines. In both cases, all units were initialized identically.

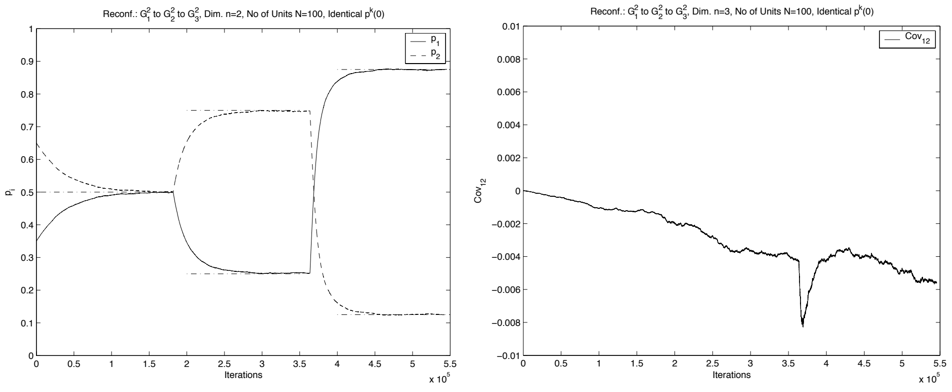

Figure 5.19: Successive system reconfigurations from $y_1^2 \to y_2^2 \to y_3^2$, N=100.

Figures (5.21), (5.22) and (5.23) show successive reconfigurations in $\Re^2$. In figure (5.21), we see a reconfiguration from $y_1^2 \to y_4^2$ where all units were identically initialized; figure (5.22) shows successive reconfigurations from $y_1^3 \to y_2^3 \to y_3^3$ from a uniform random initialization whereas in figure (5.23), the system reconfigures from $y_4^3 \to y_2^3 \to y_1^3$ after starting out far from the centroid.

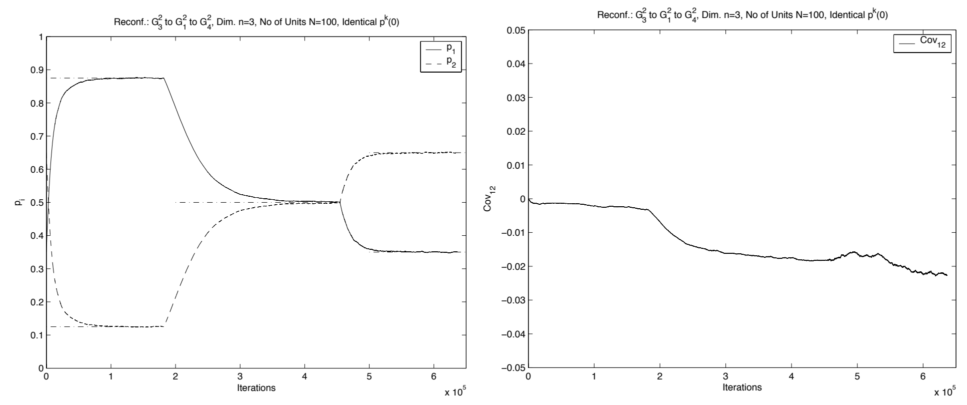

Figure 5.20: Successive system reconfigurations from $y_3^2 \to y_1^2 \to y_4^2$, N=100.

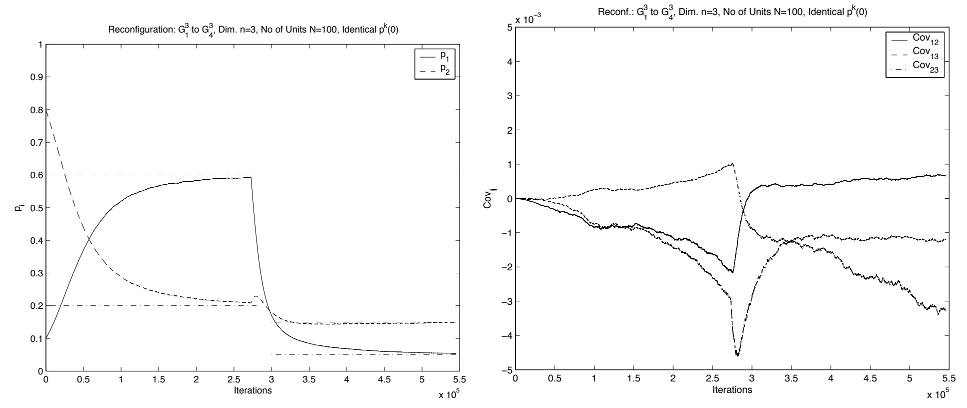

Figure 5.21: System reconfiguration from $y_1^3 \to y_4^3$, N=100.

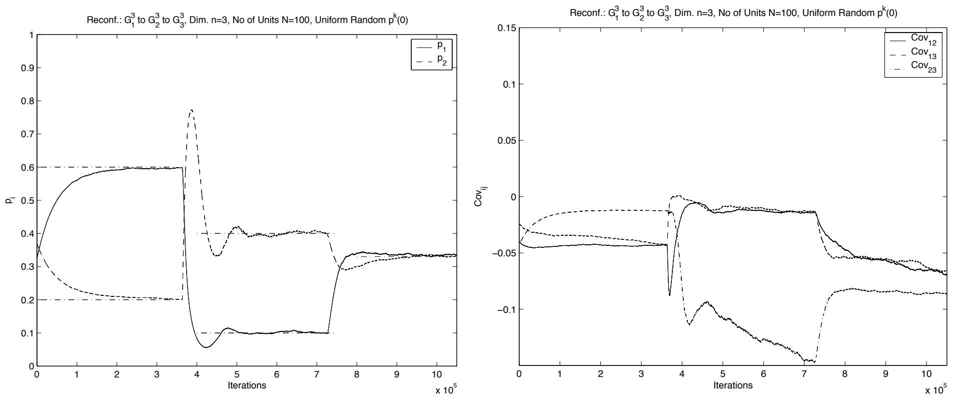

Figure 5.22: Successive system reconfigurations from $y_1^3 \to y_2^3 \to y_3^3$, N=100.

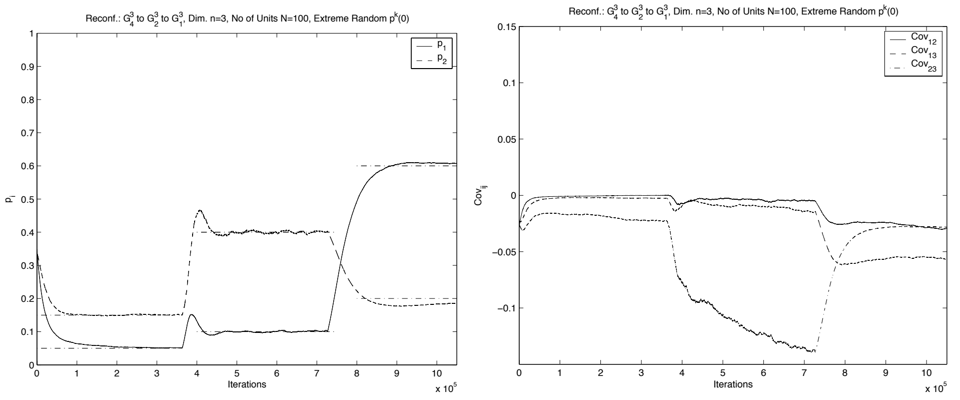

Figure 5.23: Successive system reconfigurations from $y_4^3 \to y_2^3 \to y_1^3$, N=100.

Dynamic Equilibrium The waypoints are dynamic equilibria in the sense that once the average position has reached the waypoint, the movement of the individual units does not stop, although the expected net movement over time of each individual unit becomes zero.

Individual Trajectories In figure (5.24), we see the individual trajectories of some nodes during the successive collective reconfigurations shown above in figures (5.19) and (5.20). Figure (5.25) shows the individual trajectories of some units as the system performs the successive reconfigurations seen in figure (5.23).

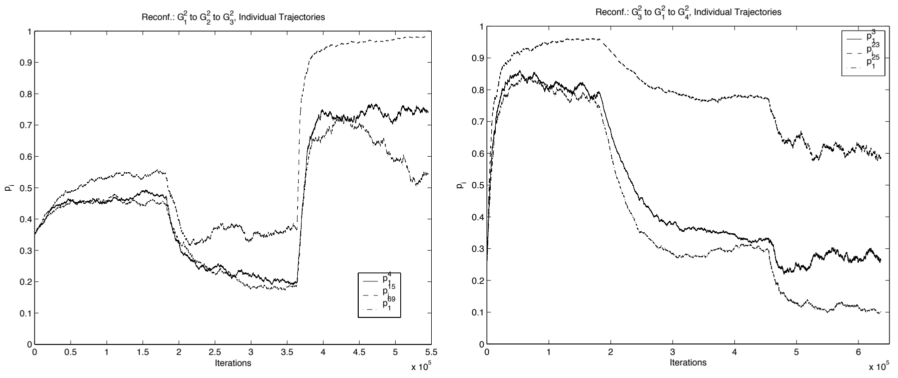

Figure 5.24: Individual trajectories of three nodes during the successive collective reconfigurations in figure (5.19), to the left, and in figure (5.20), to the right.

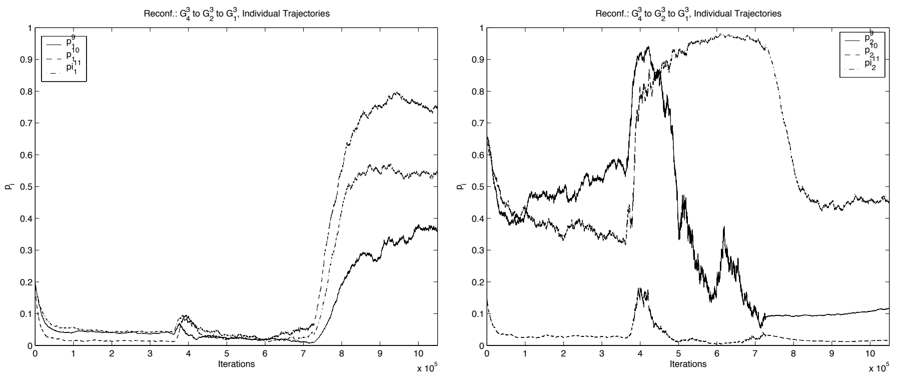

Figure 5.25: Individual trajectories of three nodes during the successive collective reconfigurations in figure (5.23).

Structure Reconfiguration We next simulated a set of structure reconfigurations, where a network of N units suffered losses of up to 50% or, conversely, was expanded by up to 50%. In either case, the change of the network size resulted in a disturbance of the average composition which varied in size with the initial network distribution. In many cases, the disturbance was negligible even with losses or additions of as much as 50%. However, when we used a biidentical initial distribution, chosen specifically to make the disturbance more noticeable, the disturbance is easily observed and followed by the readaptation of the network to the equilbrium average position.

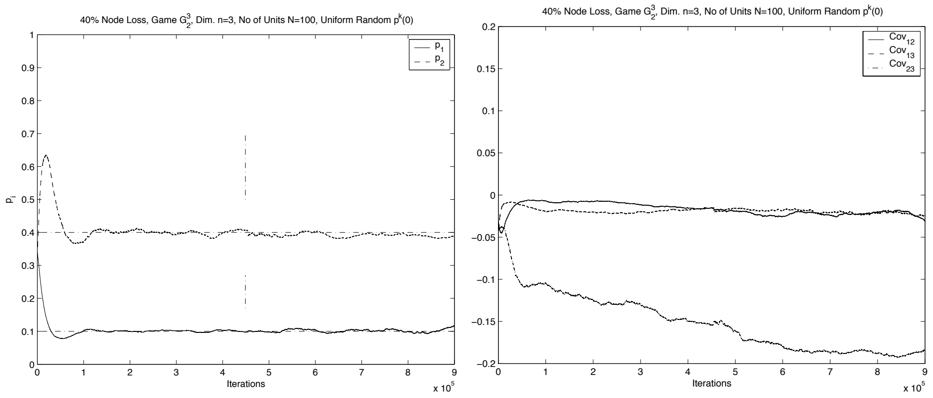

Figure 5.26: 40% Node loss at time indicated by vertical bars, game G_2^3, $N = 100$, uniform random distribution.

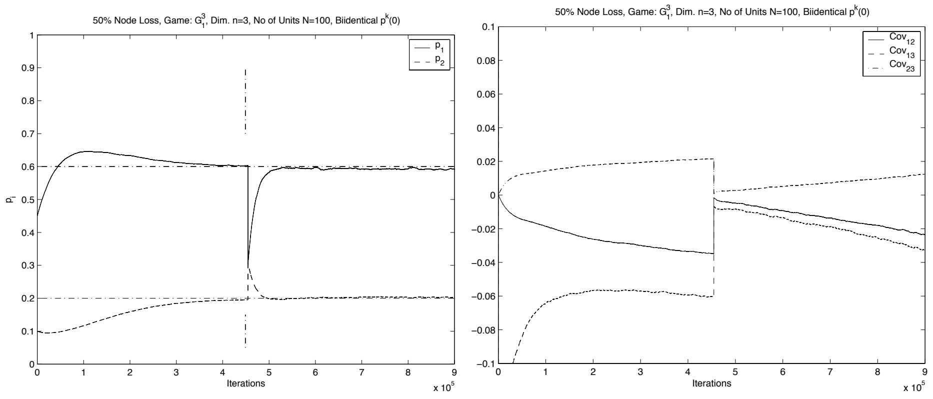

Figure 5.27: 50% Node loss at time indicated by vertical bars, game G_1^3, $N = 100$, biidentical initial distribution.

In figure (5.26), we see a 40% node loss, reducing the number of units from $N = 100$ to $N = 60$. The initial distribution being uniform random, the disturbance is barely noticeable and needs little adaptation to compensate.

In figures (5.27), (5.28) and (5.29), the initial distribution is biidentical, meaning that half of the units were given one initial distribution and the other half a different initial distribution. In figure (5.27), 50% of the original $N = 100$ nodes are lost, in figure (5.28), the network is enlarged by 50% from $N = 67$ to $N = 100$ whereas in figure (5.29), 40% of the nodes are lost, reducing the network from $N = 100$ to $N = 60$.

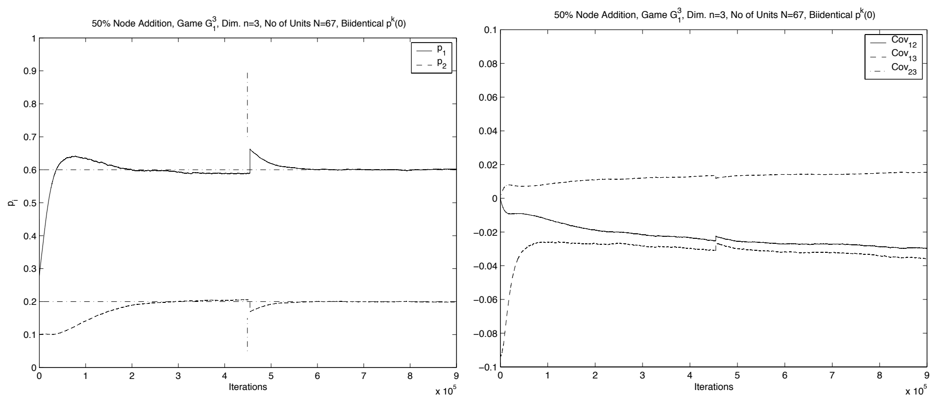

Figure 5.28: 50% Node addition at time indicated by vertical bars, game G_1^3, $N = 100$, biidentical initial distribution.

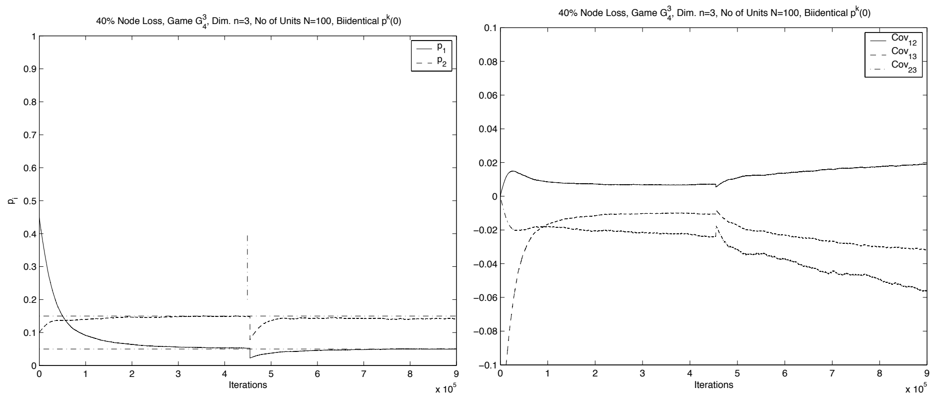

Figure 5.29: 40% Node loss at time indicated by vertical bars, game G_4^3, $N = 100$, biidentical initial distribution.

Permutation Reconfiguration The final reconfiguration category to be simulated is the permutation reconfiguration, where individual nodes switch positions while at the global level, the system stays at the same waypoint. In this scenario, we assigned to nodes $1-30$ the role of heterogeneous nodes with $y_{opt} = y_3^2$, then performed a permutation reconfiguration by reassigning this role to nodes $71-100$ and letting the first thirty nodes be homogeneous nodes.

To the left in figure (5.30), we see the convergence of the average position to the waypoint y_1^2 with a noticeable disturbance at the start of the permutation reconfiguration indicated by vertical bars. To the right in the same figure, we see the individual trajectories of three nodes, each one representing one of the three qualitatively different node groups numbered $1-30$, $31-70$ or $71-100$.

In figure (5.31), we see to the left the individual node positions just before the reconfiguration, and, to the right, the final individual node positions. As expected, to the left in figure (5.31), nodes $1-30$ are all aligned close to y_3^2 whereas the other nodes are distributed to compensate for the heterogeneous nodes. To the right in the same figure, we see that the heterogeneous role has been taken over by nodes $71-100$ and that three qualitatively different node categories have now emerged: nodes $1-30$, which were initially heterogeneous, nodes $31-70$, that were homogeneous nodes all the time and compensated for the others and, finally, nodes $71-100$, which started out homogeneous but became heterogeneous after the reconfiguration.

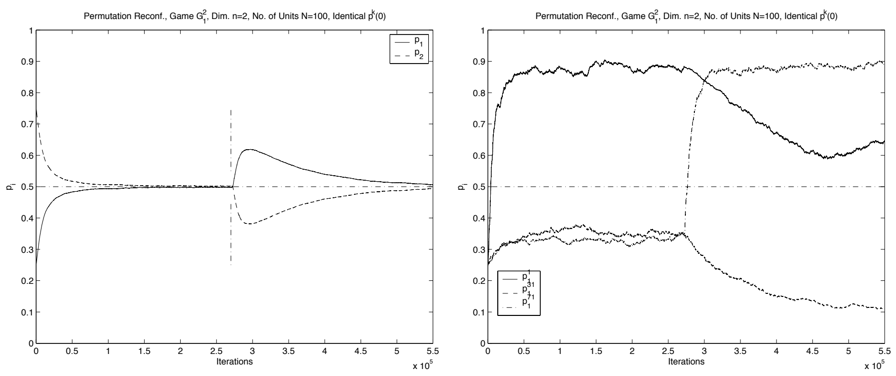

Figure 5.30: Convergence to y_1^2 with permutation reconfiguration at time indicated by vertical bars, to the left; corresponding individual trajectories respresenting three qualitatively different node categories to the right.

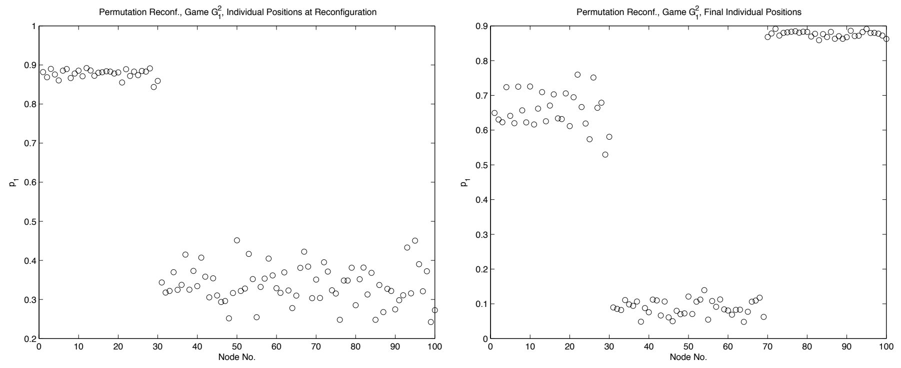

Figure 5.31: Individual node positions just before the permutation reconfiguration, to the left; final individual node positions to the right.

5.7.6 Non-Convergence Without an ESS

While all 2×2 matrices do have an ESS, that is not the case for $l \times l$ matrices where $l > 2$ - for comparison, we used a game matrix $\mathbf{G}_5^3$ that has no ESS and studied the system behavior. The average position did not converge to a waypoint for this case but oscillated around the unique Nash equilibrium $\mathbf{p^{Nash}} = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$, as can be seen in figure (5.32).

$$\mathbf{G_5^3} = \begin{array}{cccc} Player1/Player2 & Action1 & Action2 & Action3 \\ Action1 & (2,2) & (3,1) & (1,3) \\ Action2 & (1,3) & (2,2) & (3,1) \\ Action3 & (3,1) & (1,3) & (2,2) \end{array}$$

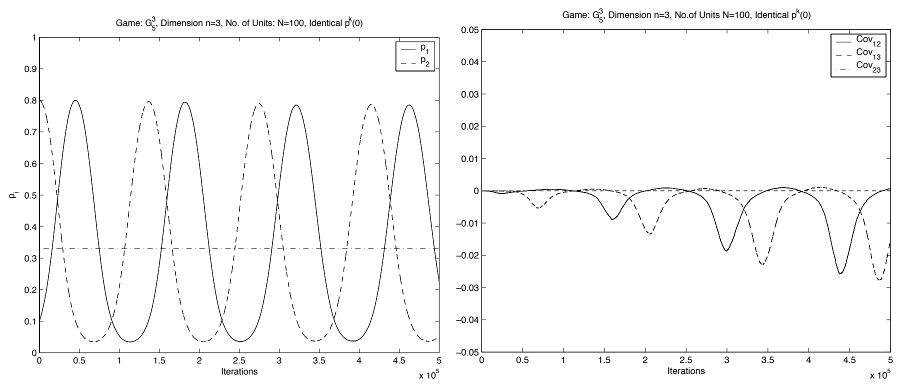

Figure 5.32: Non-convergence with $\mathbf{G}_5^3$ which has no ESS, N=100; compare with the convergence to $\mathbf{y}_3^3$ shown above in figure (5.9).

5.7.7 Different Spread for the Same ESS

In this section, we will see how the step size, that is, the learning rate influences the spread of the final individual positions around the waypoint. With a high learning rate, the convergence is fast but the final individual positions vary a good deal, wheras a low learning rate gives slower convergence but less spread. We again used the waypoint $y_1^2 = (1, \pi)$ on the unit circle; in figure (5.33), we see the convergence to y_1^2 with the same matrix scaling used above and the resulting final individual positions of the nodes. The matrix was then further scaled down by 0.1, resulting in an increase of the time for convergence by a factor of order ten that can be seen to the left in figure (5.34); to the right in the same figure, we see the corresponding final individual positions of the units, now much less spread out.

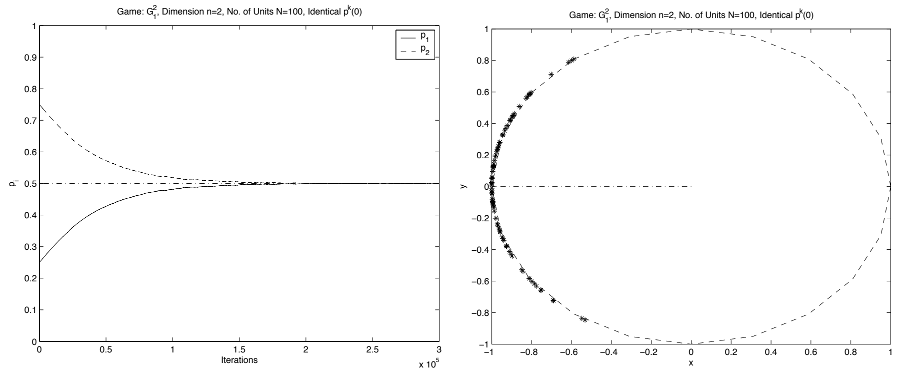

Figure 5.33: Convergence to y_1^2 to the left with higher learning rate, corresponding final individual positions to the right, $N = 100$.

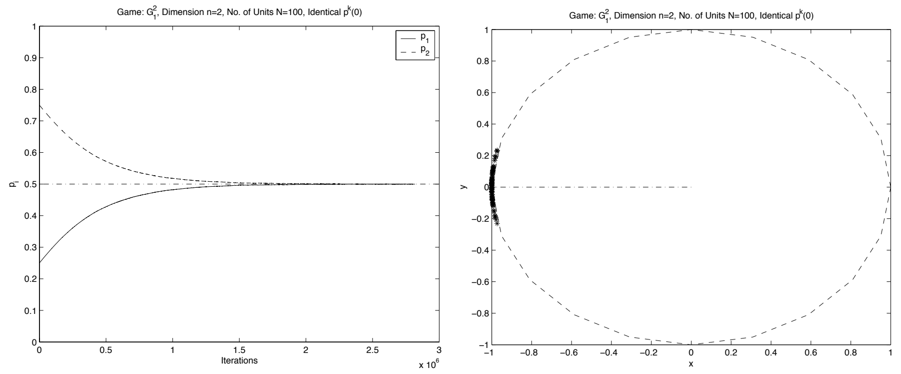

Figure 5.34: Convergence to y_1^2 to the left with lower learning rate, corresponding final individual positions to the right, $N = 100$.

Chapter 6

Cluster

In this section, we introduce the second scenario to which we propose to apply our game theoretic approach. This is a traditional formation control setting where a number of vehicles travel in formation and are required to switch formations without collision. We first give a problem statement, introduce the corresponding game and prove that in our set of chosen games, the Nash equilibrium is in each game unique. We then present a novel total field approach to collision avoidance which enables the units to navigate safely without knowing the positions of the other units. Finally, simulation results are presented along with a description of our hardware experiments.

6.1 Problem Statement

Assuming a system of N vehicles $k = 1, ..., N$, each associated at time $t = 0, 1, 2, ...$ with a position $r^k(t)$, a start position $r^k(0)$ and a goal position r^k_G in a common coordinate system R, we want all vehicles to reach their respective goal positions within a common time limit $t \leq t_{max}$ while travelling in potentially changing formations without colliding with any of the other vehicles - a type scenario is shown in figure (6.1). These positions are updated according to the same equation, where Δr is some function and $G(t)$ is a set of parameters:

$$r^k(t+1) = r^k(t) + \Delta r(r^k(t), \bar{r}(t), G(t)), \text{ where}$$

$$\bar{r}(t) = \tfrac{1}{N} \textstyle\sum_{i=1}^{N} r^i(t)$$

Recently Attracting Attention The problem of coordinating the navigation of a large number of vehicles so that they all reach their goal positions without colliding has attracted a great deal of attention lately. As the number N of vehicles grows, the problem rapidly becomes intractable with centralized approaches. Local approaches have been proposed based on nearest-neighbor interactions and/or potential fields. However, many of these approaches assume that each vehicle knows which are its nearest

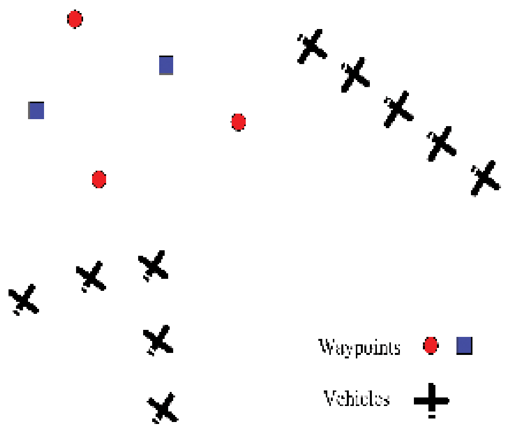

Figure 6.1: Type scenario for our problem, where a set of vehicles travel in formation and are matched to a set of waypoints to visit. How can one ensure the safe navigation of each vehicle to its waypoint(s) without collisions with any of the other vehicles when no vehicle knows the positions of the other vehicles?

neighbors and what are their positions. In fact, it may even be necessary for each vehicle to know the positions and velocities of all the other vehicles for such approaches to work.

First Local Approaches The first local approaches suggested date back to 1987 when Reynolds heuristically achieved flocking of so-called boids with simple local rules based on the positions and velocities of the vehicle's nearest neighbors [41]. A similar fixed-velocity version was proposed by Vicsek [49] in 1995.

Analytical Approaches The promising simulation results of such heuristic methods soon inspired other researchers to approach the problem analytically with the intention to actually prove the emergence of flocking as the result of applying simple local rules. The dominating approach has used potential functions and is represented by McInnes [36], Leonard [31] and Passino [20]. The flocking is here obtained as the balance of a far-field attraction and a near-field repulsion between vehicles which gives a simple expression for the resulting collective dynamics - Lyapunov stability theory can then be used to prove asymptotic stability. While theoretically elegant, these approaches make the assumption that all vehicles have or can obtain perfect information about the positions of all the other vehicles - if this is not the case, the approaches collapse. In a large scale dynamic network, this is not a realistic assumption since it would require a large scale communication intranet with frequent updates and interagent coordination and unnecessarily expose position information to those not part of the network.

Limited Information Thus, if the early heuristic studies made us believe that a local approach was possible and the subsequent approaches actually proved convergence, the next step to take is to find a scheme that is actually consistent with large, fluid swarms and makes realistic assumptions about the information available to each vehicle about the system state at any time. Indeed, this is a difficult problem in which finding the appropriate control may be quite a challenge; however, the reward of its solution will be a distributed approach that is directly applicable to the mechatronic hardware currently available.

Local Approach Necessary Why is this a hard problem? Firstly, it is a difficult problem from a control point of view since there are many degrees of freedom and the complexity grows exponentially with N. As noted above, this soon makes the problem untractable with centralized methods - a distributed control approach is necessary. This aspect of the problem is the one most often recognized as difficult in the previous literature - as noted above, the solutions proposed consisted in going from centralized control to local methods. These local methods were at first heuristic but have lately been complemented with analytical versions proven to converge [39].

Imperfect Information Another difficult aspect of the problem has to do with information and has hitherto been given little attention. While the distributed methods proposed in the last decade made the problem tractable, they kept the assumption of perfect information used in centralized control. However, such an assumption is unnatural in a distributed framework for several reasons. Firstly, such extreme redundancy of information goes against the distributed control paradigm itself - some information redundancy is indeed desired, but the duplication of all information is not. Secondly, perfect information in swarms is extremely difficult to implement with current sensing technology - estimating positions and velocities with machine vision and sonar is difficult even when N is small; as N grows, this becomes even harder since vehicles may block each other or distances to units farther away may be relatively large. Thirdly, an implementation would thus have to involve an extensive communication network which would be unnecessarily costly and might expose information, thus constituting a security risk.

Potential Field Approach? Some researchers have already attempted to find potential field schemes alternative to the perfect information approaches cited above, for problems ranging from more static single-vehicle settings [19] to the coordination of a network of vehicles [47]. In the former case, an electromagnet was used to guide an underwater vehicle to its static dock; in the latter case, inspired by insect navigation methods, artificial pheromones were introduced, the increasing or decaying quantities of which were stored in data bases and used for trajectory planning.

6.2 General System Dynamics

In this scenario, each vehicle has three different tasks to fulfill in parallel: it should travel from a start position to a goal position, keep its position in the formation and avoid colliding with any of the other vehicles, which might be a risk particularly during reconfiguration. We can think of the function Δr introduced above as the weighted sum of the three corresponding components Δr_{goal}, Δr_{conf} and Δr_{coll}. The last component, aimed at collision avoidance, should be nonzero when an obstacle - static or in the form of another navigating vehicle - is too close; the first two components, aimed at navigation and formation keeping, can be separate or combined in one single function.

6.2.1 Navigation

The task of navigation from start to goal is in formation control as a rule accomplished using either of the following three approaches [3]:

Leader-Follower In the leader-follower approach, the majority of units have no information about the goal and simply track the motion of a leader or of a weighted sum of the positions of several leaders. In this hierarchical approach, only a few units - the leaders - know where the goal is relative to their own positions and move towards it. The leaders have nonzero Δr_{goal} components whereas for the followers, navigation towards the goal and formation keeping is achieved by one single component $\Delta r_{goal/conf}$.

Tracking the Mass Center In another approach, denoted the virtual structure approach, all units track the mass center which is shifted along a desired trajectory towards the goal. This assumes a hierarchical component since the shift has to be imposed either by an exterior coordinator or by one of the nodes - the majority of the units will then automatically move along the desired trajectory by aiming to keep the distance to the mass center equal to their respective reference distances; we note that this approach fuses the tasks of navigation and formation keeping. In this case, all units have a single $\Delta r_{goal/conf}$ component for both navigation towards the goal and formation keeping and receive an update of the system mass center as a signal from the exterior coordinator.

Weighting Modes A third approach consists in letting all units be informed of the position of the goal relative to themselves and in introducing a weighting between several behavior modes, one of which is individual navigation towards the goal. In this case, there are thus separate, nonzero Δr_{goal} and Δr_{conf} components for all units.

6.2.2 Configuration Keeping

The objective of configuration keeping is achieved by tracking reference vehicles or reference points and is to at least some extent fused with the navigation towards the goal.

Leader-Follower In the leader-follower approach, the leaders do not participate in the configuration keeping - instead, the other units adapt to the leaders by aiming at keeping fixed positions relative to the moving leaders. In this way, the followers will automatically also follow a trajectory towards the goal. As a rule, each vehicle tracks a function of only one or a few of the other vehicles, which may create chains of secondary followers tracking primary followers.

Tracking the Mass Center Achieving formation keeping by letting all vehicles track the mass center is similar to the leader-follower approach; however, in this case, each vehicle tracks a function of the positions of all the other vehicles, giving a more robust situation.

Weighting Modes By weighting between navigation towards the goal and keeping a fixed distance to a moving reference point in the form of a vehicle position or a mass center, a more parallel system is created since now all units share the tasks of navigation towards the goal and formation keeping.

6.2.3 Collision Avoidance

As a third component, finally, vehicles need collision avoidance not to collide with each other or with obstacles.

Redundant for Configuration Keeping Existing approaches typically assume all vehicles to have quite advanced sensor systems that keep them informed of the positions of the vehicles to track and of whether any vehicle comes to close. Collisions between vehicles are essentially avoided by a synchronization of speeds and starting out with a sufficient spacing between vehicles.

Essential for Reconfiguration However, as soon as the issue of reconfiguration comes into the picture, the risk of collisions becomes very real indeed - the role of leader may switch to another vehicle and vehicles need to pass each other to change relative positions. Thus, an approach that works well for configuration keeping may no longer be valid in the more challenging situation of reconfiguration control. Indeed, most studies do not address reconfigurations or study only reconfigurations between similar configurations where, for example, the leader is the same.

6.3 Configuration Definition

The formation control setting typically assumes a finite, predefined set of configurations; the given application decides what configurations are relevant - for a distributed telescope, they may be circle-shaped formations of varying radii whereas for a fleet of unmanned aerial vehicles, straight lines or V-shapes may be the desired formations.

General Definition In this setting, the formations, that is, configurations are sets of node positions relative to the system mass center
$\bar{r}(t) = \frac{1}{N}\sum_{k=1}^{N} r^k(t)$, where r^k are the entries of the node position vector $\mathbf{r}$.

Corresponding Configurations Using the configuration definition introduced above, the entries $f_C^k(\mathbf{r})$ of $\mathbf{f_C}(\mathbf{r})$ are here

$$f_C^k(\mathbf{r}) = r^k - \frac{1}{N}\sum_{k=1}^{N} r^k - x_C^k,$$

where x_C^k is the desired position of node k in configuration C in a coordinate system X which has its origin at $\bar{r}$ and whose coordinate axes are parallel to and pointing in the same direction as those of R;
$F_C = \{\mathbf{0} \in \Re^N\}$, or if some imperfection is tolerated,
$F_C = \{\mathbf{z} \in \Re^N | \ |\mathbf{z}| < \epsilon\}$ for some $\epsilon > 0$.

Specific Definition For illustration, we will use the configurations
$j = 1, 2, 3, 4$, defined by the following positions relative to a local reference frame with the origin at the system mass center:

V-shape $x_1^1 = (-2.0, -0.8)$ $x_1^2 = (-1.0, 0.2)$ $x_1^3 = (0.0, 1.2)$
$x_1^4 = (1.0, 0.2)$ $x_1^5 = (2.0, -0.8)$

I-shape $x_2^1 = (0.0, -2.0)$ $x_2^2 = (0.0, -1.0)$ $x_2^3 = (0.0, 0.0)$
$x_2^4 = (0.0, 1.0)$ $x_2^5 = (0.0, 2.0)$

U-shape $x_3^1 = (-1.0, -0.8)$ $x_3^2 = (-1.0, 0.2)$ $x_3^3 = (0.0, 1.2)$
$x_3^4 = (1.0, 0.2)$ $x_3^5 = (1.0, -0.8)$

M-shape $x_3^1 = (-2.0, -1.0)$ $x_3^2 = (-1.0, 1.0)$ $x_3^3 = (0.0, 0.0)$
$x_3^4 = (1.0, 1.0)$ $x_3^5 = (2.0, -1.0)$

These formations correspond to a V-shape, an I-shape, a U-shape and an M-shape, respectively and are shown in figure (6.2).

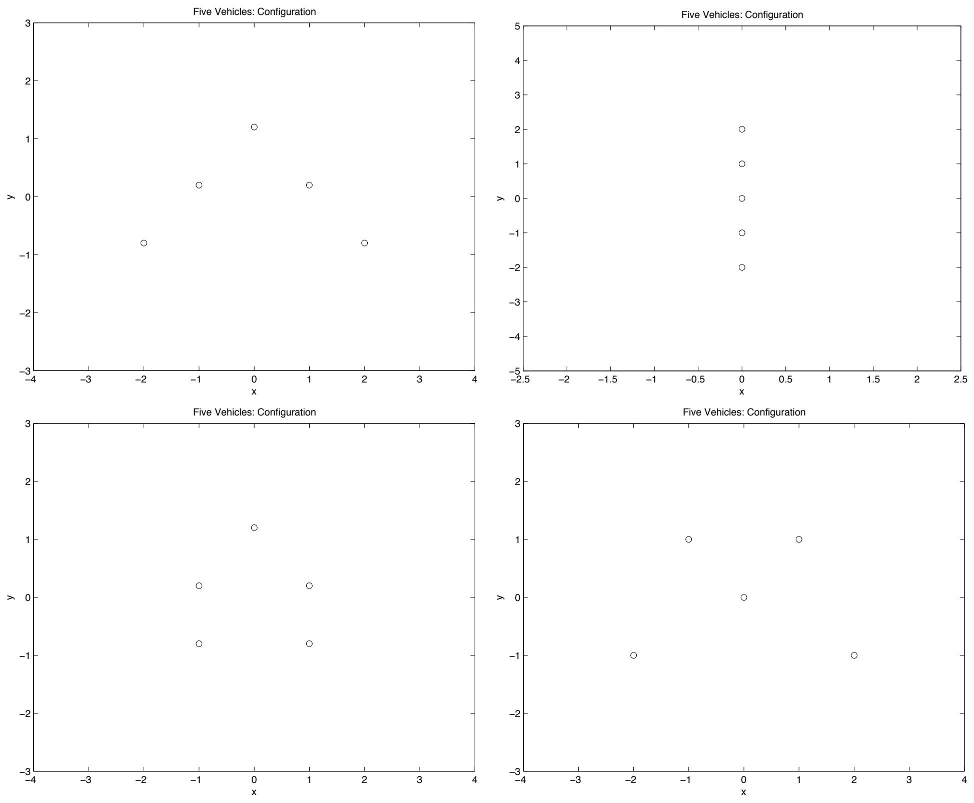

Figure 6.2: Four distinct configurations: a V-shape, an I-shape, a U-shape and an M-shape.

6.4 Equilibrium Notion: Nash Equilibrium

The equilibrium notion in this setting will be the regular Nash equilibrium rather than one of its refinements. Here, the uniqueness of the equilibrium will be a consequence of how the proposed game was constructed rather than a property of the equilibrium class itself, as was the case above. Below, we first define the game and then show that any configuration corresponds to a Nash equilibrium whereas any set of points which is not a configuration is not a Nash equilibrium.

6.4.1 Defining the Game

We define the game by naming the players, their action spaces, game parameters and payoffs.

N-player Game In the replicator learning scenario, we were mainly interested in the average position rather than the individual node positions and, although we let $N \to \infty$, for this purpose it was sufficient to use any k-player game with $k < \infty$ - we chose $k = 2$. In this scenario, however, the position of each node counts and therefore each node participates in the game each time the game is played; thus, in this case we will have an N-player game rather than a two-player game.

Strategies In addition to the common coordinate system R, we will introduce another reference frame X which has its origin at the mass center $\bar{r}(t)$ of the system of vehicles and whose coordinate axes are parallel to and pointing in the same direction as those of R. In X, the position of each vehicle k at time t will be denoted $x^k(t)$. The game will be designed as an N-player game where the strategy of each vehicle is its current position in R while its payoff is a function of its current position in X.

Parameters For each configuration j defined, each vehicle k is associated with a reference vector x_j^k in X corresponding to its assigned position in formation j.

Payoffs The payoff to each node k is defined as $-|x^k(t) - x_j^k(t)|$; we note that the payoff to any node k is ≤ 0 and that it attains its maximum when node k occupies its reference position.

6.4.2 Configuration = Nash Equilibrium

We first show that any configuration, corresponding to a set of such reference positions, is a Nash equilibrium of the game defined above.

Proof We know from above that any set of positions $(r^1(t), ..., r^N(t))$ satisfying $x^k(t) = r^k(t) - \bar{r}(t) = x_j^k$ for $k = 1, .., N$ will give the maximal payoff 0 to each player. We choose one such set and fix all strategies except that of player i.

We know that $r^i(t) - \bar{r}(t) = r^i(t) - \frac{1}{N}\sum_{k=1}^{N} r^k(t) = x_j^i$.
Any alternative strategy $y^i(t)$ can be written as $y^i(t) = r^i(t) + \delta$, where $\delta \neq 0$. This strategy will give the payoff

$$\begin{aligned} & -|y^i(t) - \frac{1}{N}\sum_{k\neq i} r^k(t) - \frac{1}{N}y^i(t) - x_j^i(t)| \\ &= -|r^i(t) + \delta - \frac{1}{N}\sum_{k=1}^{N} r^k(t) - \frac{1}{N}\delta - x_j^i(t)| \\ &= -|\frac{N-1}{N}\delta + 0| = -\frac{N-1}{N}|\delta| < 0 \end{aligned} \tag{6.1}$$

Since this is true for every node, the configuration is a strict Nash equilibrium.

6.4.3 Out of Configuration = No Nash Equilibrium

We will next show that no point set $(r^1(t), ..., r^N(t))$ can be a Nash equilibrium of the game unless it satisfies $-|x^k(t) - x_j^k(t)| = 0$ for each $k = 1, ..., N$.

Proof The proof is by contradiction. We assume that $(r^1(t), ..., r^N(t))$ is a Nash equilibrium and that for at least one i, $-|x^i(t) - x_j^i(t)| < 0$.
This means that there exists an $\alpha \neq 0$ such that
$r^i(t) - \frac{1}{N}\sum_{k=1}^{N} r^k(t) - x_j^i(t) - \alpha = 0$. However, this implies that the alternative strategy $y^i(t) = r^i(t) - \frac{N}{N-1}\alpha$ gives a strictly higher payoff to cell i than $r^i(t)$, since

$$\begin{aligned} & -|y^i(t) - \frac{1}{N}\sum_{k\neq i} r^k(t) - \frac{1}{N}y^i(t) - x_j^i(t)| \\ &= -|r^i(t) - \frac{N}{N-1}\alpha - \frac{1}{N}\sum_{k=1}^{N} r^k(t) + \frac{1}{N}\frac{N}{N-1}\alpha - x_j^i(t)| \\ &= -|\alpha - \frac{N}{N-1}\alpha + \frac{1}{N}\frac{N}{N-1}\alpha| = -|\frac{N-1-N+1}{N-1}\alpha| = 0 > -|\alpha|. \end{aligned} \tag{6.2}$$

Thus, $(r^1(t), ..., r^N(t))$ is not a Nash equilibrium.

6.5 Proposed Algorithm

Our proposed algorithm consists of two separate parts: firstly, the game theoretic analysis presented above according to which each configuration is seen as the unique Nash equilibrium of a different game and, secondly, a total field collision avoidance algorithm that permits safe navigation without requiring the vehicles to know each other's positions.

Idea In Brief

We propose a total field sensing approach of magnetic nature to ensure collision avoidance as an alternative to approaches where each vehicle needs to know the positions of some or all of the other vehicles in order to plan a safe trajectory.

Configuration as Unique Nash Equilibrium While in most formation control approaches, vehicles use the actual positions of other vehicles or the mass center of all or some vehicles as reference points in their navigation, the collective goal of the system is merely the sum of the individual tracking goals of each vehicle - no formal system theoretic view is offered. As described above, we propose a game theoretic view of the collective system goal, seeing the current formation as the unique equilibrium of a game completely described by its parameters and by interpreting switches of formations as switches of such games.

Total Field Approach In this setting, each vehicle generates a magnetic field around itself and is provided with a set of magnetic sensors that sense the total field generated by the whole network of vehicles. The vehicle's own field is static in its reference frame and known; however, the field contributions of the other vehicles are unknown and vary as the vehicles move. Since the magnetic flux density decays inversely proportional to the cube of the distance from the magnetic source, the contribution of vehicles farther away will be very small compared to that of vehicles situated closer to the sensing vehicle. The magnetic field generated by the sensing vehicle itself can be cancelled out with more or less sophisticated methods, leaving as the net result sensed the dynamic field generated by the other vehicles.

Strategic Sensing In particular, by strategically placing one-axis magnetic sensors orthogonal to the vehicle's own field at different locations, the vehicle's own field will not be included in the measurements. Furthermore, by choosing these locations in a symmetric way, one can also automatically cancel out Earth's magnetic field component, thus making sure that the net difference only reflects the total field generated by the $N-1$ other vehicles. By combining the measurements of several on-board sensors placed at different locations, the vehicle can estimate the gradient of the total field generated by the other vehicles and move in the opposite direction to avoid collision. Thus, no vehicle has to know the position of any of the other vehicles in order to plan its trajectory. An interesting application where the total field sensing approach could be useful is robosoccer [16].

Natural Systems Analogy Recent results in biology suggest that similar mechanisms may be found in natural systems such as flocks of birds, schools of fish and bee swarms [18]. Indeed, it has been verified that the navigation skills of such systems can be severely hampered by the application of disturbing magnetic fields.

Superposition on Naive Approach As noted above, centralized approaches to the problem rapidly become intractable as the number of vehicles N grows and it is therefore necessary to opt for a decentralized approach, making each vehicle responsible for its own trajectory planning. Ideally, each vehicle would be able to navigate along a straight line from its start position to its goal without colliding with other vehicles - our approach consists in superposing on such a naive scheme a collision avoidance algorithm based on total field sensing which makes information about the particular positions and velocities of the other vehicles unnecessary. Since this is a highly dynamic form of potential field, we will first briefly review the theory of artificial potential fields in trajectory planning.

6.5.1 Potential Fields: Background

The potential field approach for obstacle avoidance, proposed by Khatib in 1986 [27], models goals and obstacles as attractors and repellers of an artificial potential field in which each vehicle moves.

On-line and Tractable The great advantage of this approach was its tractability, which allowed highly adaptive on-line trajectory planning instead of the rigid, time-consuming but exact off-line approaches in use at the time. The most serious shortcoming of the approach was the risk of local minima and of oscillations near obstacles and in narrow passages [28].

Perfect Information The theoretical simplicity of the potential field approach comes at the price of assuming perfect information or perfect extraction of information from the sensors. In the former case, the navigating vehicle has a map, knows the positions of all the obstacles relative to itself and can thus inversely deduce the forces that it would sense. In the latter case, the vehicle has sensors good enough to perfectly estimate the positions and velocities of the obstacles - the sensing modes traditionally suggested are machine vision or sonar - and then again deduce the forces. The first case implies a static situation with no real need for on-line path planning whereas in the second case, unrealistic assumptions are made on the sensors. In both cases, the unnatural detour needed to actually obtain the ficticious forces goes against the simplicity of the theoretical framework.

Other Sensing Modes? Is it possible to try other sensing modes to actually sense real forces, thus giving the potential field approach a simplicity in implementation to match that of the theoretical framework? Could the sensing of such forces be made more accurate than the estimation of distances and velocities based on visual or sonar input?

Real Force We thus want to find a common signal, preferably in the form of a real force, that can be readily measured and informs each vehicle if any of the other vehicles is close and in which direction it is approaching. Gravity and electromagnetic forces all decay with the distance between the source and the sensor, which is desirable in our case. While gravity and electrostatic forces seem inappropriate due to weak signals and lack of suitable sensors, respectively, magnetic forces can conveniently be measured with a range of commercial sensors.

Magnetism With this in mind, we propose a total field collision avoidance scheme of magnetic nature which requires each vehicle to generate a local magnetic field with an on-board magnet and sense the total surrounding field with a set of magnetic sensors.

6.5.2 Magnetic Field Theory: Background

Charged particles in motion generate magnetic fields - here, we review the central notions in magnetic field theory and examine the flux density around current-carrying conductors in general and the magnetic dipole in particular.

Definitions The two central notions in magnetic field theory are the magnetic flux density B and the magnetic field intensity H, both vector quantities. We will deal with the magnetic flux density B, given in teslas, T, or in gauss, 1 gauss $= 10^{-4}$ T - the magnetic field of Earth corresponds at its surface to a magnetic flux density component of about 0.5 gauss.

In figure (6.3), we see an illustration of the magnetic field generated by a small bar magnet at the origin; to the left in the figure, we see the direction of the field whereas to the right, the field magnitude is shown.

Magnetic Material Permanent magnets generate a constant static field around them, the strength of which is determined by the magnitude and direction of their magnetic moments. This, in turn, depends on the material of which the magnet is made and on its size and shape. Electromagnets, on the other hand, are created by winding a current-carrying conductor in many coils around cores of magnetic material and switching on the current.

Current-Carrying Conductor The magnetic flux density around an infinitely long circular current-carrying conductor forms concentric circles in any plane perpendicular to the direction of the current. When the conductor

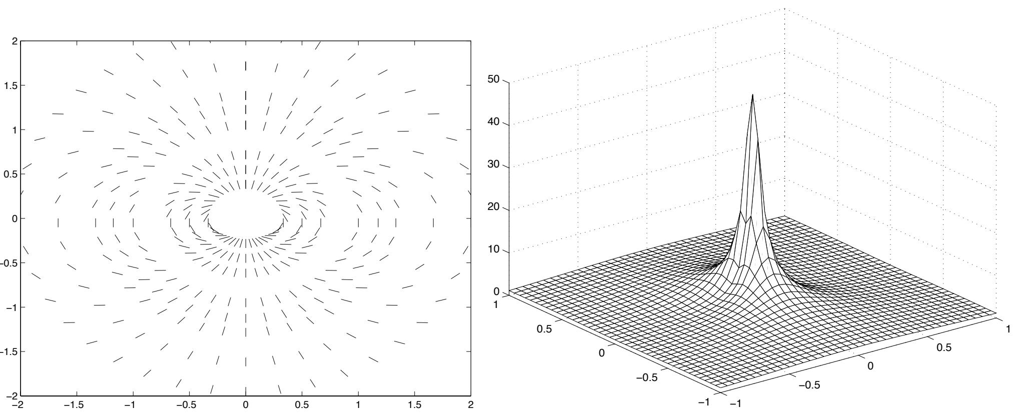

Figure 6.3: Magnetic flux density $\frac{B}{|B|}$ in the xy-plane around a magnetic dipole, to the left, and $|B|$, to the right.

has finite length and forms a closed loop, we obtain what is referred to as a magnetic dipole.

Magnetic Dipole The magnetic flux density **B** around a magnetic dipole - a small current-carrying loop - can be described as

$$\mathbf{B} = \frac{\mu_0 m}{4\pi r^3}(\mathbf{e}_r 2\cos\theta + \mathbf{e}_\theta \sin\theta), \tag{6.3}$$

where $\mu_0 = 4\pi \cdot 10^{-7}$ H/m is the permeability of free space, $m\mathbf{e}_y$ is the magnetic dipole moment, with $\mathbf{e}_y$ as given in figure (6.4), and $(r,\ \theta)$ are the spherical coordinates in a coordinate system that has the dipole at its origin, also as shown in figure (6.4) [11, 34]. The vectors $\mathbf{e}_r$ and $\mathbf{e}_\theta$ are the orthogonal unit vectors in the spherical coordinate system. Importantly, the magnetic flux lines around a bar magnet can be very well approximated by those generated by a magnetic dipole.

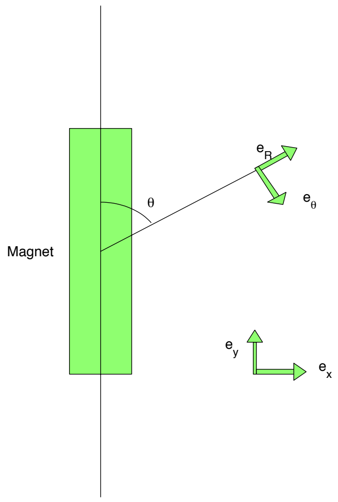

Figure 6.4: Spherical coordinate axes and Cartesian coordinate axes.

6.5.3 Vehicles as Magnetic Dipoles

By providing each vehicle with a magnet, we transform our system of navigating vehicles into a system of mobile magnetic dipoles that can be sensed by each vehicle. Each vehicle generates a magnetic field around itself and has onboard magnetic sensors to sense the sum of the magnetic fields of the other vehicles, thus obtaining information about how close the other vehicles are.

Two issues need to be addressed in particular: firstly, how to make the vehicles not sense their own magnetic field, only that of the other vehicles and, secondly, how to exclude the magnetic field of Earth.

6.5.4 Generating the Local Field

Each vehicle k carries a magnet of moment m on board that generates a magnetic field $\mathbf{B}^k(\cdot,t)$. Although constant in the local reference frame R^k of vehicle k, the magnetic field $\mathbf{B}^k(\cdot,t)$ will vary in the common reference frame R as vehicle k moves. By the analogy with a magnetic dipole described above, each magnet naturally defines a local spherical coordinate system with the magnet at the origin. At any point j described in R^k by the coordinates (r_k^j, θ_k^j), the field $\mathbf{B}^k(j,\cdot)$ generated by vehicle k will be

$$\mathbf{B}^k(j,\cdot) = \frac{\mu_0 m}{4\pi (r_k^j)^3}(\mathbf{e}_{r_k^j} 2\cos\theta_k^j + \mathbf{e}_{\theta_k^j}\sin\theta_k^j). \tag{6.4}$$

6.5.5 Sensing the Total Field

In addition to the magnet, each vehicle has a set of magnetic field sensors on board with which it can sense the total magnetic field at the corresponding set of points and draw conclusions about the gradient of the total magnetic field. From above, we have that the total field sensed at a point j by any vehicle will be

$$\mathbf{B}(j,\cdot) = \sum_{k=1}^{N} \frac{\mu_0 m}{4\pi (r_k^j)^3} (\mathbf{e}_{r_k^j} 2\cos\theta_k^j + \mathbf{e}_{\theta_k^j} \sin\theta_k^j). \tag{6.5}$$

As we can see, the cubic factor in the denominators above will make the vehicle's own magnetic field the dominating term in the sum above when j is the position of an on-board sensor.

6.5.6 Subtracting Own Field

Since the vehicle's own magnetic field is constant and known, ideal sensors would make it possible to subtract this field and obtain the net total field generated by the $N-1$ other vehicles. Each vehicle k would then sense the other vehicles at point j as $\mathbf{B}_S^k(j)$, where

$$\mathbf{B}_S^k(j,\cdot) = \sum_{l \neq k} \frac{\mu_0 m}{4\pi (r_l^j)^3} (\mathbf{e}_{r_l^j} 2\cos\theta_l^j + \mathbf{e}_{\theta_l^j} \sin\theta_l^j). \tag{6.6}$$

Certain commercial sensors have patented offset straps which make it possible to apply a magnetic field in the direction opposite to the external magnetic field, thus cancelling out the external field and making sure that the net field stays in the range within which the sensing is linear. This thus permits a more sophisticated form of subtraction of the own field.

6.5.7 Strategic Sensor Positions

Due to hardware limitations and the difference in magnitude between the vehicle's own field and that generated by the other vehicles, a straightforward subtraction as described above may not always be possible. However, using the fact that the vehicle's own field is constant and known both in magnitude and direction in the vehicle's own reference frame, one can make the described subtraction unnecessary by strategically positioning the sensors to measure the total field orthogonal to the vehicle's own field at different points.

As noted above, at a given on-board point j the sensor senses the vehicle's own field as $\mathbf{B}^k(j,\cdot)$, where r_j^k is the distance from the magnet to the sensor and θ_j^k the angle between the axis of the magnet and the vector from the magnet to the sensor:

$$\mathbf{B}^k(j,\cdot) = \frac{\mu_0 m}{4\pi (r_k^j)^3} (\mathbf{e}_{r_k^j} 2\cos\theta_k^j + \mathbf{e}_{\theta_k^j} \sin\theta_k^j). \tag{6.7}$$

Since all magnetic field lines are closed and continuous, it is possible to find exact angles θ_j^k where the vehicle's own field is respectively parallel to or orthogonal to the axis of the vehicle's magnet. The former case is the easier one: when θ_j^k is respectively 0, $\frac{\pi}{2}$ or π, the vehicle's own field is parallel to the magnet axis. The latter case requires some calculations and leads to the values for θ_j^k of $\arccos\frac{1}{\sqrt{3}}$ or $(\pi - \arccos\frac{1}{\sqrt{3}})$, that is, 54.7^o or 125.3^o.

To prove this, we first express the spherical unit vectors $\mathbf{e}_r$ and $\mathbf{e}_\theta$ in the Cartesian unit vectors $\mathbf{e}_x$ and $\mathbf{e}_y$, where $\mathbf{e}_y$ is parallel to the magnet axis, as shown in figure (6.4). We obtain, with the sign depending on the quadrant:

$$\mathbf{e}_r = \pm\sin\theta\mathbf{e}_x + \cos\theta\mathbf{e}_y \tag{6.8}$$
$$\mathbf{e}_\theta = \pm\cos\theta\mathbf{e}_x - \sin\theta\mathbf{e}_y \tag{6.9}$$

Using these relations, we now have:

$$2\cos\theta\mathbf{e}_r + \sin\theta\mathbf{e}_\theta = 2\cos\theta(\pm\sin\theta\mathbf{e}_x + \cos\theta\mathbf{e}_y) + \sin\theta(\pm\cos\theta\mathbf{e}_x - \sin\theta\mathbf{e}_y)$$
$$= \pm\,3\cos\theta\sin\theta\mathbf{e}_x + (2\cos^2\theta - \sin^2\theta)\mathbf{e}_\theta \tag{6.10}$$

For the first case, we want the x-component of the field, $\pm 3\cos\theta\sin\theta$, to be zero. We have:

$$\pm 3\cos\theta\sin\theta = 0 \Rightarrow \cos\theta = 0 \text{ or } \sin\theta = 0 \Rightarrow \theta = 0, \frac{\pi}{2}, \pi. \tag{6.11}$$

In the second case, we want the y-component, $2\cos^2\theta - \sin^2\theta$, to vanish. Using the fact that $\cos^2\theta + \sin^2\theta = 1$, we have:

$$2\cos^2\theta - \sin^2\theta = 2\cos^2\theta - 1 + \cos^2\theta = 3\cos^2\theta - 1 = 0$$
$$\Rightarrow \cos\theta = \frac{\pm 1}{\sqrt{3}} \Rightarrow \theta = \arccos\frac{\pm 1}{\sqrt{3}}. \tag{6.12}$$

We have now found a set $\Theta_x = \{0, \frac{\pi}{2}, \pi\}$ of angles where the vehicle's own field is orthogonal to the x-axis in the vehicle's reference frame and a set of angles $\Theta_y = \{\arccos\frac{1}{\sqrt{3}}, (\pi - \arccos\frac{1}{\sqrt{3}})\}$ where, likewise, the vehicle's own field is orthogonal to the y-axis in the vehicle's reference frame; these positions are shown in figure (6.5). Thus, if a field component along the x-axis is detected at an angle $\theta_x \in \Theta_x$ or a field component along the y-axis is measured at an angle $\theta_y \in \Theta_y$, we know that they were not generated by the vehicle itself.

6.5.8 Earth's Magnetic Field

The magnetic field of Earth is generated by magnetic material in the deep interior of the planet. At any point on the Earth's surface, the field has a component in the tangent plane directed towards the magnetic South pole, close to the geographical North pole, and an orthogonal component which we disregard in a two-dimensional setting.

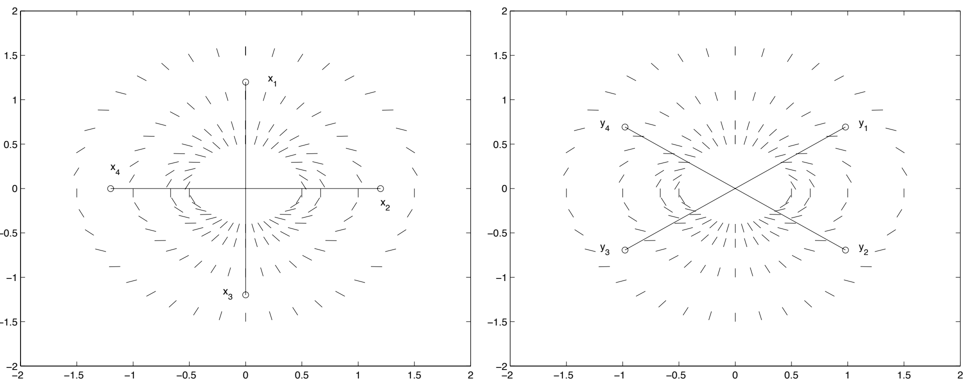

Figure 6.5: Sensor input positions for measuring the surrounding field along the x-axis, to the left, and along the y-axis, to the right.

While the strategic sensor positions described above ensure that the vehicle's sensors will not sense its own field, the total field sensed will in general include a component of the magnetic field of the Earth - as the vehicles rotate, this component will change. However, the symmetric choice of the strategic sensor positions as illustrated above will make sure that the component $\mathbf{B}_x$ of the Earth's magnetic field sensed by any of the sensors positioned to measure the x-component will be identical, and the same is true for the component $\mathbf{B}_y$ sensed by the sensors positioned to measure the y-component. Thus, the pairwise subtractions of the measurements of x-sensing sensors or of y-sensing sensors will always cancel out the Earth magnetic field component.

Why will the magnetic field component of Earth measured by the sensors be the same while that is not true for the measurements of the total field generated by the $N-1$ other vehicles? That is because the distance to the magnetic field source of Earth is infinitely much larger than the distances to the other vehicles, so that the on-board sensors of a vehicle are all at essentially the same distance from the interior of Earth whereas their respective distances to the vehicles vary and make a difference.

6.6 Proposed System Dynamics

In order to navigate towards the goal while avoiding collisions during possible reconfigurations, each vehicle must use its sensor measurements to estimate the gradient of the total field generated by the other vehicles to get information about what directions to avoid.

Weighted Mode We will use the weighted mode approach where collision avoidance, navigation towards the goal and configuration keeping are weighted to obtain the desired net result of safe, yet aggressive navigation towards the goal.

6.6.1 Navigating towards the Goal

Naive Approach When $N = 1$, there is no uncertainty in the system and we can let our solitary vehicle navigate along the straight line from $\mathbf{r}^1(0)$ to $\mathbf{r}^1_G$. However, as soon as $N > 1$, such naive paths may no longer be safe and detours will have to be planned around detected vehicles. In addition, as the number of vehicles N increases, the uncertainty grows exponentially and the task of keeping track of the exact positions of all or even a fraction of the other vehicles becomes overwhelming - rather, one will have to focus on whether any of the other vehicles is too close.

Superposition Safe navigation towards the goal is achieved as the weighted sum of collision avoidance and the movement of the system mass center along the straight line towards the goal. We have for some weight γ, $0 < \gamma < 1$:

$$\Delta r_{goal} = \gamma \frac{\bar{r}_G - \bar{r}}{|\bar{r}_G - \bar{r}|}. \tag{6.13}$$

6.6.2 Configuration Keeping

Configuration keeping is obtained by making each node keep its distance to the system mass center, $r^k(t) - \bar{r}(t)$, equal to its current reference vector x^k_j.

$$\Delta r_{conf} = \gamma \frac{r^k - \bar{r} - x^k_j}{|r^k - \bar{r} - x^k_j|}. \tag{6.14}$$

6.6.3 Collision Avoidance

The collision avoidance component Δr_{coll} is parallel to the estimated gradient of the total field ∇B_{est}, to be described next, but of opposite direction. We thus have:

$$\Delta r_{coll} = -(1 - \gamma) \frac{\nabla B_{est}}{|\nabla B_{est}|}. \tag{6.15}$$

6.6.4 Estimating the Gradient

Each vehicle has eight single-axis sensors positioned at the eight strategic positions chosen above. The four sensors positioned to sense the x-component of the surrounding magnetic field are numbered clockwise and give the inputs x_i, $i = 1, 2, 3, 4$ corresponding to the spherical angles 0^o, 90^o and 180^o in the left part of figure (6.5). The same is true for the four sensors measuring the y-component as inputs y_i, $i = 1, 2, 3, 4$; they correspond to the spherical angles 54.7^o and 125.3^o in the right part of figure (6.5).

We wish to estimate the gradient $\nabla B = \frac{\partial}{\partial x} B\mathbf{e_x} + \frac{\partial}{\partial y} B\mathbf{e_y}$ at the origin of the vehicle's own reference frame. We have

$$\frac{\partial}{\partial x} B = \frac{\partial}{\partial x}\sqrt{B_x^2 + B_y^2} = \frac{1}{\sqrt{B_x^2 + B_y^2}}(B_x \frac{\partial B_x}{\partial x} + B_y \frac{\partial B_y}{\partial x}) \text{ and} \tag{6.16}$$

$$\frac{\partial}{\partial y} B = \frac{\partial}{\partial y}\sqrt{B_x^2 + B_y^2} = \frac{1}{\sqrt{B_x^2 + B_y^2}}(B_x \frac{\partial B_x}{\partial y} + B_y \frac{\partial B_y}{\partial y}). \tag{6.17}$$

We approximate

$$\frac{\partial B_x}{\partial x} \approx \frac{x_2 - x_4}{l_{x_{24}}} \text{ and } \frac{\partial B_x}{\partial y} \approx \frac{x_1 - x_3}{l_{x_{13}}}, \tag{6.18}$$

where $l_{x_{ij}}$ is the distance between the sensors x_i and x_j.
We average to approximate

$$\frac{\partial B_y}{\partial x} \approx \frac{y_1 - y_4}{2l_{y_{14}}} + \frac{y_2 - y_3}{2l_{y_{23}}} \text{ and } \frac{\partial B_y}{\partial y} \approx \frac{y_4 - y_3}{2l_{y_{43}}} + \frac{y_1 - y_2}{2l_{y_{12}}}, \tag{6.19}$$

where $l_{y_{ij}}$ is the distance between the sensors y_i and y_j.
The values B_x and B_y are estimated from the sensor measurements as

$$B_x \approx \frac{x_1 + x_2 + x_3 + x_4}{4} \text{ and } B_y \approx \frac{y_1 + y_2 + y_3 + y_4}{4}. \tag{6.20}$$

The estimate $\nabla_{est} B_{est}$ of the gradient thus obtained is used to generate a collision avoidance component in the motion of the vehicle. We have

$$\begin{aligned}\nabla_{est} B_{est} = & \frac{1}{\sqrt{B_x^2 + B_y^2}}(B_x \frac{x_2 - x_4}{l_{24}} + B_y(\frac{y_1 - y_4}{2l_{y_{14}}} + \frac{y_2 - y_3}{2l_{y_{23}}}))\mathbf{e_x} \\ & + \frac{1}{\sqrt{B_x^2 + B_y^2}}(B_x \frac{x_1 - x_3}{l_{13}} + B_y(\frac{y_4 - y_3}{2l_{y_{43}}} + \frac{y_1 - y_2}{2l_{y_{12}}}))\mathbf{e_y},\end{aligned} \tag{6.21}$$

where B_x and B_y are approximated as in (6.20).

6.6.5 Net Motion

A third important factor is the weighting between the two possibly conflicting objectives of orientation towards a goal and collision avoidance, respectively. We wanted to achieve aggressive path planning and got the best results with a weight of $\gamma = 0.66$ on the orientation towards the goal and $(1-\gamma) = 0.34$ on collision avoidance. The net motion is described by Δr as given below.

$$\begin{aligned}\Delta r &= \Delta r_{goal} + \Delta r_{conf} + \Delta r_{coll} && (6.22)\\ &= \gamma \frac{r_G - r}{|r_G - r|} + \gamma \frac{r^k - \bar{r} - x_j^k}{|r^k - \bar{r} - x_j^k|} - (1-\gamma)\frac{\nabla B_{est}}{|\nabla B_{est}|} && (6.23)\end{aligned}$$

Choosing Weight γ The weight γ needs to be chosen large enough to ensure that the vehicles reach their destination, yet small enough that they do not collide. By definition of the system dynamics in reconfiguration when $\Delta r_{conf} = 0$,

$$r^k(t+1) = r^k(t) + \gamma \frac{r_G^k - r^k(t)}{|r_G^k - r^k(t)|} - (1-\gamma)\frac{\nabla B}{|\nabla B|}. \tag{6.24}$$

Defining the function $L^k(t) = |r^k(t) - r_G^k|$, we thus have

$$\begin{aligned}L^k(t+1) &= |r^k(t+1) - r_G^k|\\ &= |r^k(t) + \gamma \frac{r_G^k - r^k(t)}{|r_G^k - r^k(t)|} - (1-\gamma)\frac{\nabla B}{|\nabla B|} - r_G^k|\\ &= |(L^k(t) - \gamma)\frac{r^k(t) - r_G^k}{|r^k(t) - r_G^k|} - (1-\gamma)\frac{\nabla B}{|\nabla B|}|\\ &\leq |(L^k(t) - \gamma)\frac{r^k(t) - r_G^k}{|r^k(t) - r_G^k|}| + |(1-\gamma)\frac{\nabla B}{|\nabla B|}|\\ &= (L^k(t) - \gamma) + (1-\gamma)\\ &= L^k(t) + (1-2\gamma)\\ &< L^k(t) \text{ iff } 1 - 2\gamma < 0 \Rightarrow \gamma > \frac{1}{2} && (6.25)\end{aligned}$$

Thus, if $\gamma > \frac{1}{2}$, $L^k(t)$ is a strictly decreasing sequence with a lower boundary $L_-^k = 0$ and a finite initial value $L^k(0) < \infty$.

6.7 Stability Analysis

To evaluate the stability of our approach, we will first analyze the resulting navigation in environments that are much simpler and more static than the scenario for which the algorithm was designed.

6.7.1 Two Vehicles

The two-vehicle situation can easily be expressed analytically; in the given equation, we note that although the distance $|r^1 - r^2|$ to the other vehicle is the same for both vehicles, the angles θ_1 and θ_2 are generally not the same.

$$\begin{bmatrix} \Delta r^1 \\ \Delta r^2 \end{bmatrix} = \gamma \begin{bmatrix} \frac{r_G^1 - r^1}{\sqrt{r_G^1 - r^1}} \\ \frac{r_G^2 - r^2}{\sqrt{r_G^2 - r^2}} \end{bmatrix} - (1-\gamma) \begin{bmatrix} \nabla \frac{\mu_0 m}{4\pi (r^2 - r^1)^3} (\sqrt{4\cos^2\theta_2 + \sin^2\theta_2}) \\ \nabla \frac{\mu_0 m}{4\pi (r^1 - r^2)^3} (\sqrt{4\cos^2\theta_1 + \sin^2\theta_1}) \end{bmatrix} \tag{6.26}$$

Qualitatively Different The two- or three-vehicle situation is qualitatively different from the multi-vehicle scenario because in the former situation, each vehicle has much more information about who generated the field it is sensing - this additional information should be used. In the general multi-vehicle situation, the field sensed gives little if any information about the positions of the individual units.

One Vehicle Static We simulated the two-vehicle case where one vehicle is static - in the left part of figure (6.6), we see how the minimum distance between a navigating vehicle and another vehicle, here assumed static, varies with γ. In the right part of figure (6.6), the trajectories are plotted for $\gamma = 0.55$, $\gamma = 0.6$ and $\gamma = 0.66$ when the orientation α of the static vehicle with respect to the x-axis is 90^o; from this study we concluded that the interesting interval for γ is quite small.

Introducing Asymmetry in Roles A straightforward approach for ensuring that each vehicle arrives at its destination without collision is to introduce an asymmetry in the form of dominance - one vehicle is assigned a dominant role and does not deal with collision avoidance at all, whereas the other vehicle is given a subordinated role and yields to the other vehicle. Once the first vehicle has reached its destination, the second vehicle will no longer have to yield and collision-free convergence is guaranteed.

Indeed, an asymmetry in roles can be generalized to the case where $N > 2$ by giving all players distinct identity numbers $k = 1, 2, ..., N$ and by letting vehicle $k = 1$ be the dominant one and the others be subordinate. Once vehicle $k = 1$ has arrived at its destination, it is broadcast that vehicle $k = 2$ takes over the dominant role - this procedure is repeated until, if vehicle $k = N$ still has not arrived, it will take on the dominant role and then in fact be alone with no need to yield.

Introducing Asymmetry in Time Another alternative which can be generalized to the case where $N > 2$ is to instead introduce an asymmetry in time, letting the units start at points close in time rather than at the same time - this approach can be used for problems with symmetrical initial conditions. Whether a system has symmetrical initial conditions can be determined by seeing if the naive approach alone would not only lead to too narrow margins but indeed also to actual collisions. In a reconfiguration setting, this would thus mean that the vehicles started the reconfiguration in series rather than at the same time.

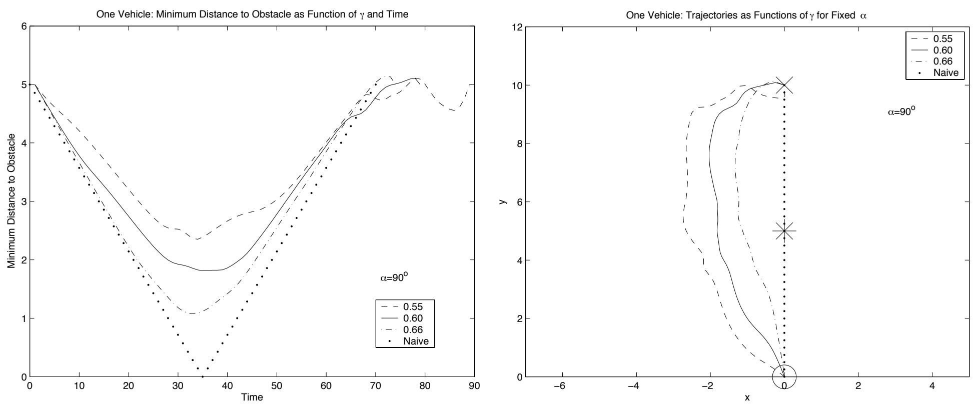

Figure 6.6: Variation with γ of the minimum distance between a navigating vehicle and a static vehicle, to the left, and three different trajectories for $\gamma = 0.55$, $\gamma = 0.6$ and $\gamma = 0.66$ when the orientation α of the static vehicle with respect to the x-axis is 90^o, to the right.

PD-regulator To increase the angular inertia, the collision avoidance part Δr_{coll} can be formed as a weighted average of the current and previous normalizations of the field gradient, as expressed in equation (6.27). A simple z-transform analysis then shows the collision avoidance part of our controller to be a PD-regulator with respect to the gradient of the total field magnitude $|B|$ as given in equation (6.29).

$$\Delta\mathbf{r}_{coll}(t) = \alpha\mathbf{e}_{\nabla}(t) + \beta\mathbf{e}_{\nabla}(t-1) \tag{6.27}$$

$$\Delta\mathbf{r}_{coll}(t) = K_P\mathbf{e}_{\nabla}(t) + K_D\frac{\mathbf{e}_{\nabla}(t) - \mathbf{e}_{\nabla}(t-1)}{2} \tag{6.28}$$

$$\alpha = K_P + \frac{K_D}{2}, \beta = -\frac{K_D}{2} \tag{6.29}$$

6.7.2 Potential Dangers

The main risks to the system stability are the presence of deadlocks, oscillations, the possibility of collisions and observability.

Static Deadlocks A static deadlock is a situation where at least one vehicle stops moving altogether, caught by opposing forces of zero resultant which on the one hand strive to move the vehicle towards its goal and, on the other hand, aim at avoiding collisions. For our algorithm, this can only occur if the weight $\gamma \leq 0.5$.

Dynamic Deadlocks In a dynamic or cyclic deadlock, on the other hand, the vehicles still move but only in a cyclic manner within a restricted space. Typically, the relative motion of the vehicles during this cycle gives rise to conflicting commands for each vehicle at different points, such as focus on collision avoidance versus navigation towards the goal. This can happen if $\gamma < 0.5$; such values of γ are thus not allowed. However, even if $\gamma < 0.5$ this risk is small whereas algorithms constructed to allow for no collisions at all, briefly described below, carry much higher risks of dynamic deadlocks.

Oscillations Oscillations are a general danger in potential field approaches - in our case, they are above all caused by low angular inertia in the vehicle model, which causes the surrounding field to vary too much between two successive points in time, thus causing the responses from the individual vehicles to be very different at these two time points.

Collisions The more aggressive the navigation, the narrower the margins and, in consequence, the higher the risk of collisions. A worst-case scenario that allows for no collisions leaves most of the benefits of the distributed paradigm unused - the fundamental strength of the distributed system is its parallel structure that allows the system to function well although fractions of the system are lost.

In some cases, safer navigation with larger margins may be preferred to aggressive maneuvering - we obtained qualitatively different and less aggressive navigation by making all vehicles orient their magnets along an agreed upon axis, thus making sure the superposition of all fields did not cancel out any part of the individual components.

Observability Low observability is often an additional constraint in actual applications, where in addition to navigating safely, vehicles should avoid detection by potential opponents, who are assumed to be able to observe everything in certain possibly moving observation fields - this can be pictured as an observation circle moving along some trajectory over the space in which the vehicles are moving.

By seeing the center of the moving observation field as a ficticious vehicle, our algorithm can easily be adapted to this case as well. Each actual vehicle takes into account not only the real fields generated by the other vehicles but also weighs in a ficticious field component as generated by the ficticious vehicle - the weight ξ can be chosen to reflect the importance attached to avoiding detection. This gives the following expression for the ficticious field $\mathbf{B}_{fict}$ and the modified surrounding field:

$$\mathbf{B}_{fict}((j,\cdot) = \xi \frac{\mu_0 m}{4\pi (r_l^j)^3}(\mathbf{e}_{r_l^j} 2\cos\theta_l^j + \mathbf{e}_{\theta_l^j}\sin\theta_l^j) \tag{6.30}$$

$$\mathbf{B}_S^k(j,\cdot) = \sum_{l\neq k} \frac{\mu_0 m}{4\pi (r_l^j)^3}(\mathbf{e}_{r_l^j} 2\cos\theta_l^j + \mathbf{e}_{\theta_l^j}\sin\theta_l^j) + \mathbf{B}_{fict}(j,\cdot) \tag{6.31}$$

We tested this approach in our original setting to which a moving observation field was added. As expected, the addition of an additional constraint made the collision avoidance margins to the real vehicles somewhat tighter while also assuring that the vehicles kept a safety margin to the center of the observation field as desired.

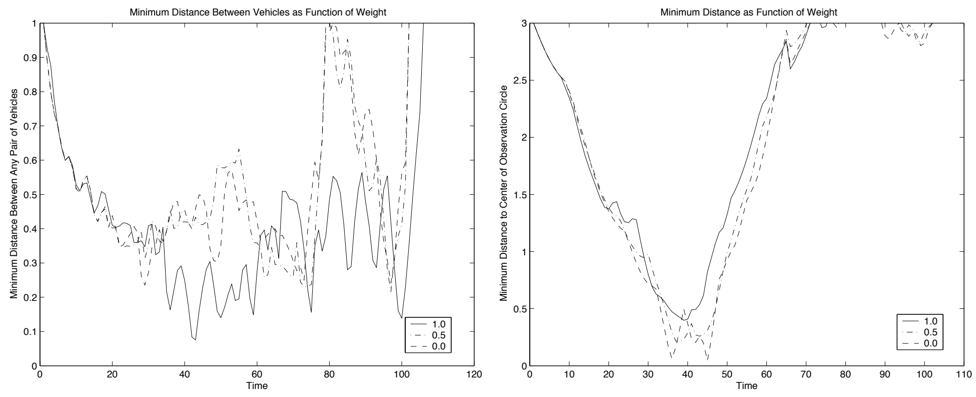

Figure 6.7: Variation with ξ of the minimum distance between any pair of vehicles, to the left; variation with ξ of the minimum distance between any vehicle and the center of the observation field, to the right.

6.7.3 Factors of Influence

We concluded above that if the weight $\gamma > 0.5$, all vehicles will indeed arrive at their respective destinations but the margins to the other vehicles may still vary with a number of factors, of which the main ones are presented below.

Weight γ If $\gamma \leq 0.5$, the total field approach in its current version can indeed give rise to static deadlocks. We found the interesting interval of γ to be small, $0.55-0.66$, as stated above - beyond this interval, the margins were too narrow or the objective of navigating towards the goal became subordinated and vehicles overreacted to the presence of other vehicles, producing oscillating trajectories and crossing each other's paths more than once.

Space Available vs. N Clearly, given a limited space there is an upper bound on the number of vehicles N that can navigate within the space without colliding - if N is too large, each vehicle will be surrounded by vehicles and face a high risk of collision no matter what direction it chooses.

Step Size vs. Vehicle Size In this scenario, the vehicles are not modelled as points but have noninfinitesimal dimensions as reflected by the maximal distance l_v between the on-board sensors. The step size relative to the vehicle dimensions is a factor to be taken into consideration - we let the step size be $1.0 - 1.5$ times the dimensions of the vehicle.

Symmetry Symmetric problems may pose a challenge by leading to mirroring trajectories along the original axis of symmetry which makes it difficult for vehicles to cross paths. The most straightforward way of solving this is to introduce an asymmetry in time, making the nodes start at slightly different points in time.

Orientation α The relative orientation $\alpha_{ij}(t)$ of two vehicles i and j at time t plays an important role in the extent to which the vehicles sense each other - with a periodicity of π, the vehicles have a blind spot. In the case where there are only two vehicles, particularly if one of them is static, this may be a problem, but in the standard case where $N > 2$, $\alpha_{ij}(t)$ depends on so many factors that it is extremely unlikely that even a static vehicle j will be in the blind spot of a vehicle i for more than one time step.

Angular Inertia Imposing a maximum turning angle $\theta_{\max}$ per time step makes the total field changes more smooth for each vehicle and thus reduces the risk for oscillations in the trajectories. However, if $\theta_{\max}$ is too small, there may be a risk of staying close to blind spots since the relative orientation $\alpha_{ij}(t)$ may hit such a spot for some combination i and j and time t.

6.7.4 Alternative Formulations

We tried some variations of the total field algorithm in which the field magnitude was taken into account and the step size or weight varied; although more elaborate, we found them to work less well than the original version of the algorithm for reasons given below.

Magnitude Rather than Gradient Currently, the gradient is normalized and the resulting displacement is the same whether the magnetic field sources are situated far away or close to the sensing vehicle - to be able to guarantee that no collisions will occur, each vehicle has to stop as soon as the field magnitude $|B|$ sensed by any sensor is above a threshold $|B|_{\max}$. If the field magnitude sensed by any of the other sensors is smaller than a minimum threshold $|B|_{\min}$, the vehicle may move in the direction of the negative gradient - such a minimum threshold is needed since if the field magnitude is very large in all directions around the vehicle, any movement may give rise to a collision. Assuming N is finite, a subset N_{bound} of all vehicles will be positioned on the boundary of the convex hull $A_N(t)$ of the area in which the vehicles are currently moving. These vehicles, at least, will have freedom of movement in the direction away from $A_N(t)$. However, the resulting movement will not have the dynamics typical of high-speed vehicles but rather be suitable for an explorative, slow rover-like vehicle.

Variable Step Size Allowing the step size to be variable would make it possible to take large steps when $|B|$ is small and small steps when $|B|$ is large. However, such a scheme is vulnerable to static deadlocks, as verified in simulations, since in a cluttered environment all vehicles may end up taking very small steps, leading to a static situation.

Variable Weight γ By letting the weighting parameter γ vary with $|B|$, one can focus on navigation towards the goal and let $\gamma \to 1$ when $|B|$ is small and, conversely, let $\gamma \to 0$ when $|B|$ is large and collision avoidance has the higher priority. Unfortunately, we found this approach to be quite vulnerable to dynamic deadlocks.

Step Back in Emergency When the field gradient is parallel or almost parallel to the direction towards the goal, it is clear that the main local obstacle region for that vehicle lies right in front of it. A slight modification to the algorithm that was heuristically shown to increase the margins to the other vehicles was to let the vehicle take a step along the negative gradient if the angle β between the field gradient and the direction towards the goal was less than a minimum value $\beta_{\min}$. This is a special case of variable weight γ since $\gamma = 0$ if $\beta < \beta_{\min}$.

6.8 Heterogeneity

In contrast to the first scenario described above, in the cluster scenario each functional role is individually designed for the corresponding node and can thus be adapted to suit its particular profile. If one node reacts slower to change than the other nodes, it may not be required to be active in permutation configurations but will keep its current relative position while the other nodes switch roles.

Reduced Position Spectrum In the first scenario, we focused on the heterogeneity where some nodes are limited in their choice of positions as compared to others; this particular type of heterogeneity is important in the cluster scenario, too. This can thus be seen as the heterogeneous players having a reduced action space compared to the other players; since the actions are positions in R and the position spectrum is defined relative to the mass center in X, the positions to avoid will change with time.

Defensive Position Preference In particular, some nodes may prefer a defensive position behind other nodes rather than occupying one of the often more energy-consuming front roles.

Relative Position Preference Some nodes may instead have position preferences defined relative to the positions of other nodes. This also corresponds to a reduced action space as compared to other nodes, but with the preferred positions now conditional on the current strategies of just a few nodes rather than on all nodes through the mass center - the whole payoff function for that particular player is thus changed, not just its parameters.

In the leader-follower approach cited above, the followers are artificially assumed to have such relative position preferences; in this analysis, such preferences will only be assumed when they correspond to a physical constraint such as two players jointly carrying one item and therefore needing to be side by side.

While such a preference may be difficult to accommodate in the general case, it easily adapts to several standard formations such as the V- or I-shapes introduced above.

Slower Dynamics If a node is impaired so that its dynamics changes, for example by slowing it down, either the network as a whole will have to slow down to stay in formation or the impaired node will have to be discarded. However, in the special case where different nodes may have different speeds, such as a satellite network forming concentric circles of different radii circling around Earth in synchronization, the impaired node may shift orbits to an orbit of lesser radius.

6.9 Reconfiguration

How are the three classes of reconfigurations introduced in the classification above - permutation, system and structure reconfigurations - interpreted in this setting?

6.9.1 Permutation Reconfiguration

Whereas in the swarm application described above, the main interest was the average position, in this high precision system the exact position of each node counts and the issue of permutation reconfigurations has higher priority. Since each individual formation position corresponds to a reference position with respect to the mass center of the system, a switch of roles corresponds to a switch of reference positions - from a game theoretic point of view, the players switch payoff functions with each other but otherwise the game is still the same.

6.9.2 System Reconfiguration

System reconfigurations are switches from one predefined formation to another - in our example for $N = 5$, between the V-, I-, U- and M-formations - and are here implemented as a synchronized switch of game parameters, that is, individual reference positions with respect to the mass center of the system. Thus, only the parameters used in the payoff functions change, not the payoff functions themselves.

6.9.3 Structure Reconfiguration

A structure reconfiguration goes beyond the mere swapping of game parameters, either at the level of local nodes or at the level of the whole network. In a structure reconfiguration, some payoff functions are replaced by new ones or the very format of the game changes such as the number of players N, requiring a redefinition of the whole set of payoff functions.

In node loss, if the formation position of a lost node is known, one can compensate for it when estimating the mass center of the system by adding a virtual node at its formation position while awaiting the arrival of a replacement node. When the formation position of the lost node is not known, it may be possible to obtain such information indirectly by analyzing the disturbance of the mass center trajectory caused by the loss of the node.

The addition of nodes will call for corresponding adjustments of the reference positions of the other nodes; the number of such adjustments should preferably be made as small as possible.

6.9.4 Reconfiguration Initiation

Reconfiguration initiation concerns reasons for reconfiguring, who takes the initiative to reconfigure and how the roles are allocated among the nodes.

Reasons for Reconfiguration

The network may have many different reasons for reconfiguring - we present four main categories of such reasons and give motivations.

Function As a rule, the network has a function, which may range from forming a distibuted telescope of varying radius to working as an elastic fence around a vessel navigating in the ocean. The function is as a rule tied to the relative positions of the nodes, thus to their configuration, and reconfigurability is usually the reason for using a network of small, simpler units rather than one single, large unit.

Environment The network is located in an environment which may change either because the network is moving or because the environment itself is dynamic. Although the network may be filling a constant function at the time, such as forming a telescope of constant radius, it may now need to reconfigure in order to fill the same function in a changing environment. For example, a navigating network may need to reconfigure from a V-shape to an I-shape to be able to pass through a narrow passage and avoid collision or it may have to reconfigure from an I-shape to a V-shape to be able to better observe the environment and not have nodes blocking each other's view.

Individual Nodes As the individual nodes change, permutation or structure reconfigurations may be necessary because an impaired node needs to be reallocated to a different role. Furthermore, permutation reconfigurations may also be called for to get a balanced energy consumption among the nodes, since some roles may be more energy-consuming than others.

Interaction with Other Networks Interaction with other networks may be yet another reason for reconfiguration; the objective may be to detect and locate the other network such as in searching for a number of static landmines or, when the other network is a mobile enemy, to avoid being observed or to assume a defensive configuration that makes it more difficult to attack.

Taking the Initiative

If each predefined configuration is identified by an index, any node can initiate a reconfiguration by broadcasting the index of the desired new configuration. However, such a scheme puts a good deal of responsibility on each node and may be vulnerable both to single errors in judgment and to interior network attacks. Furthermore, a period of saturation during which no messages can be broadcast needs to follow to avoid conflicting reconfiguration commands.

In an alternative approach, different nodes may be responsible for initiating reconfigurations depending on the current configuration - this would avoid the possibility of conflicting reconfiguration commands.

Yet another approach is to have a central coordinator, either exterior or part of the network, that receives distributed sensing input from the network and makes the final decision about reconfiguration, possibly on the recommendation of local nodes.

Role Allocation Once the decision has been made to reconfigure, each node needs to know its role in the new configuration. One alternative is to give each role in every configuration a number j, $j = 1, ..., N$ and to let each node k, $k = 1, ..., N$ always have the role j, $j = k$. At reconfiguration, the nodes are informed of the new mass center and the new configuration and find their way to their new relative positions with the total field approach.

Role Assignment By Priority If heterogeneous nodes are present that have a reduced position spectrum, they may be given priority in choosing their next roles, given that a new configuration has been chosen. The rest of the nodes may then be allowed to choose their respective new roles in order of increasing index k.

Asynchronous Reconfiguration Since many configurations have at least one axis of symmetry and the total field approach is sensitive to symmetries when N is small, it may be a good idea to let the nodes start reconfiguring at points close in time rather than simultaneously.

6.10 Simulation Results

In this section, we present a series of simulations carried out to test our approach. Formation keeping has already been extensively studied using the three different approaches cited above - leader-follower, virtual structure or weighted mode - and we therefore focus on the collision avoidance and reconfigurations.

6.10.1 Collision Avoidance

In this scenario, ten vehicles A, B,...,J were assigned start and goal positions from the sets S and G of start and goal positions, respectively, where $S = \{(i, 0) \mid i = 0, 1, ..., 9\}$ and $G = \{(i, 10) \mid i = 0, 1, ..., 9\}$. All vehicles started at different points $s \in S$ and had different goal points $g \in G$. Although this scenario was constructed to study collision avoidance in general, we node that it can also be interpreted as a permutation reconfiguration of an I-shape composed of $N = 10$ vehicles.

To evaluate the performance of the algorithm, the minimum distance between any pair of vehicles was plotted as a function of time both for the total field approach with $\gamma = 0.66$ and for the naive approach with no collision avoidance - the result is shown in figure (6.8). While the naive scheme with no collision avoidance indeed led to collisions, the total field approach made sure each vehicle kept a minimal safety distance to all other vehicles.

In figures (6.9), (6.10) and (6.11), we see comparisons between trajectories generated by the total field approach and the naive scheme, respectively. At some particularly interesting points in time, the simultaneous positions of all ten vehicles are marked with the symbols o, * or Δ.

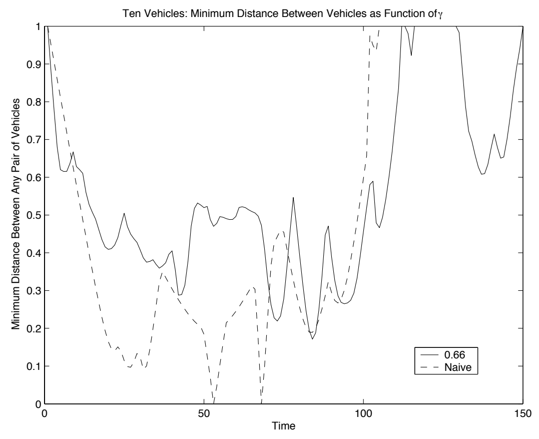

Figure 6.8: Minimum distance between any vehicle pair, total field approach and naive approach without collision avoidance compared; $\gamma = 0.66$ for the total field approach.

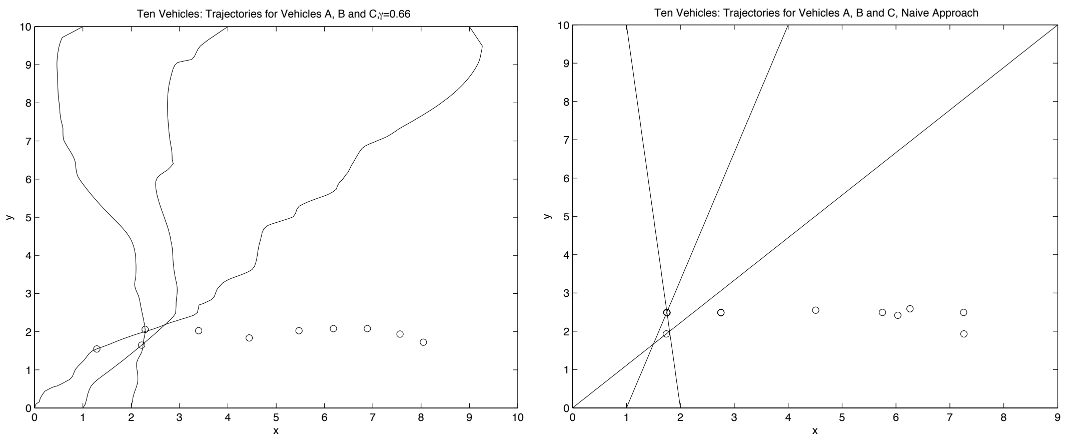

Figure 6.9: Safe aggressive trajectories for vehicles A, B and C, to the left; trajectories without collision avoidance to the right. The positions of all vehicles at a particular time are indicated by the symbol o.

6.10.2 Reconfiguration

Reconfigurations of all the types identified above - permutation, system and structure reconfigurations - were also simulated.

System Reconfiguration In figures (6.13), (6.14), (6.15) and (6.16), we see system reconfigurations between the four configurations introduced above. In figure (6.13), the system reconfigures from the V-shape to the I-shape; in figure (6.14), it goes from the V-shape to the U-shape and rotates simultaneously, whereas in figure (6.15), we see a reconfiguration from the I-shape to the U-shape. Figure (6.16), finally, shows a reconfiguration from the V-shape to the M-shape with simultaneous rotation.

The reconfiguration trajectories are shown to the left in all figures, while to the right we see how the minimum distance between vehicles changes over time during the reconfiguration with the total field approach in solid line and a standard potential field approach dashed; the initial formation is shown with double circles whereas the final formation is indicated with simple circles.

The naive scheme with no collision avoidance would indeed lead to collisions in figures (6.15) and (6.16), but the total field approach gave safe trajectories. Analyzing the situation in figure (6.15), we see that the collision would be caused by the symmetry of the subproblem for nodes 4 and 5, whereas in figure (6.16), the crossing paths of vehicles 3, 4 and 5 rather than symmetry would be to blame.

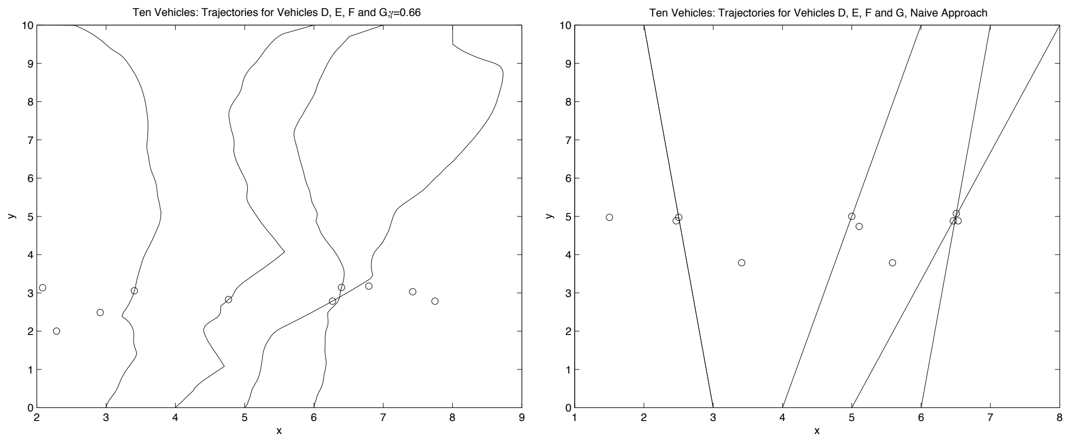

Figure 6.10: Safe aggressive trajectories for vehicles D, E, F and G, to the left; trajectories without collision avoidance to the right. The positions of all vehicles at a particular time are indicated by the symbol o.

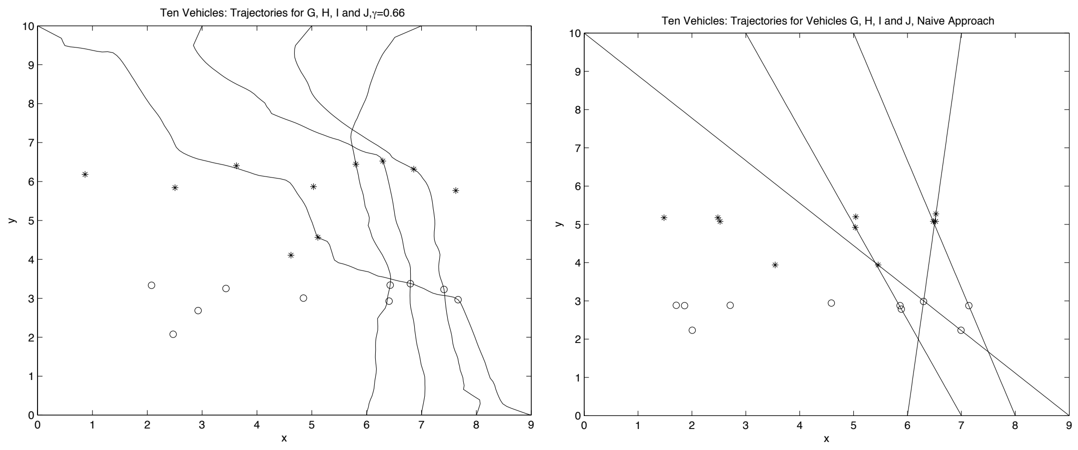

Figure 6.11: Safe aggressive trajectories for vehicles G, H, I and J, to the left; trajectories without collision avoidance to the right. The positions of all vehicles at two particular times are indicated by the symbols o and *, respectively.

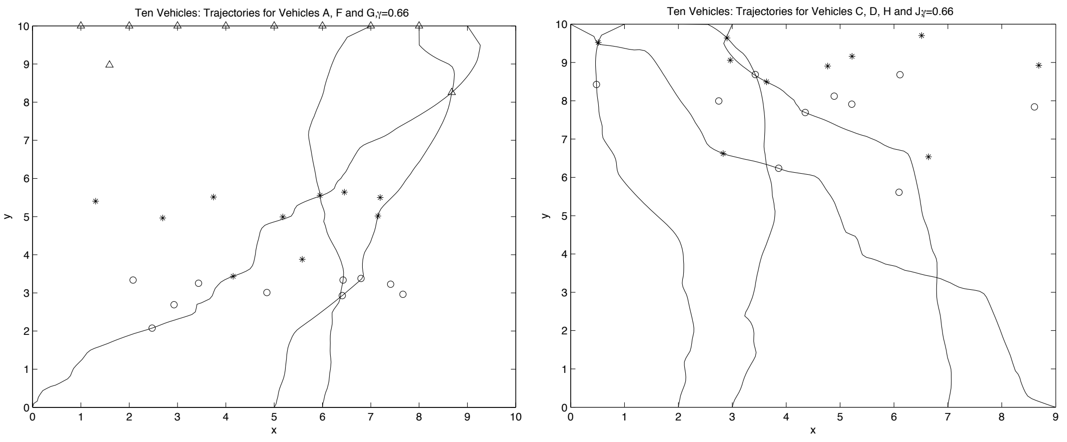

Figure 6.12: Safe aggressive trajectories for vehicles A, F and G, to the left, and for C, D, H and J, to the right. The positions of all vehicles at three particular times are indicated by the symbols o, * and Δ, respectively.

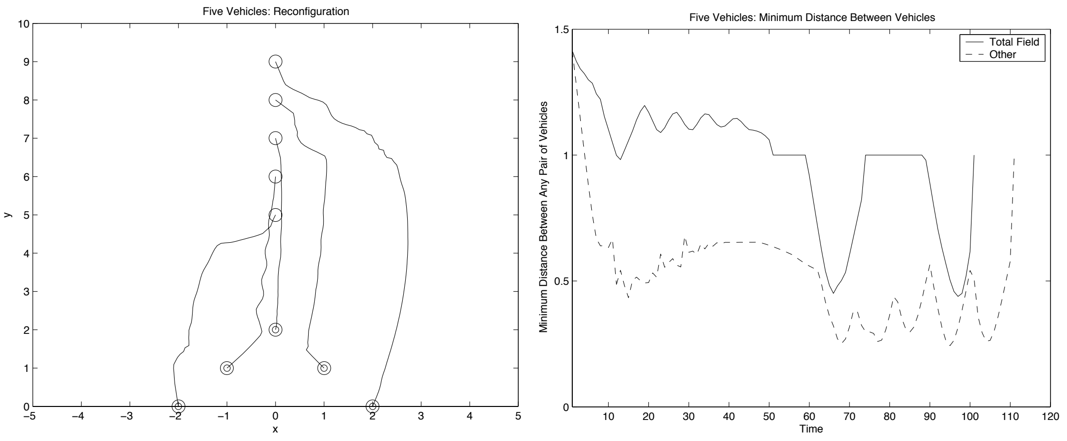

Figure 6.13: System reconfiguration between the V-shape and the I-shape, to the left, and the minimum distance between vehicles during the reconfiguration, total field approach and potential field approach compared, to the right.

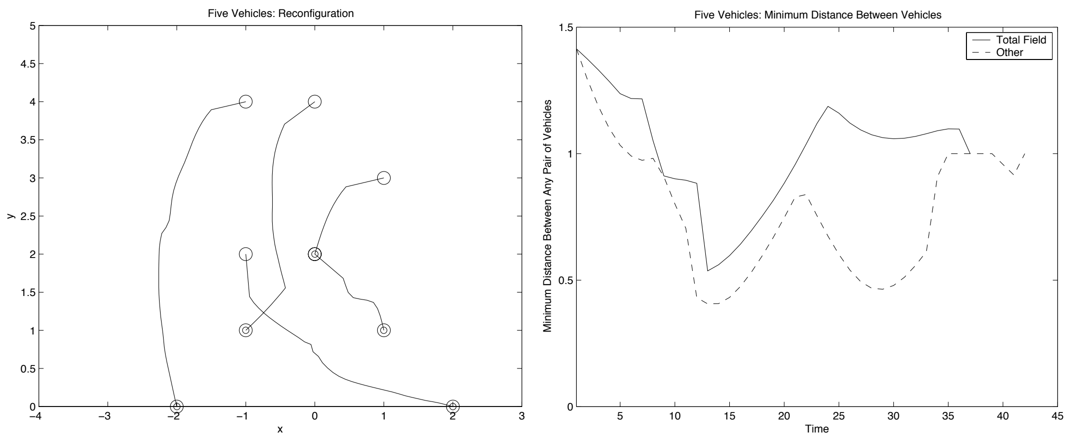

Figure 6.14: System reconfiguration between the V-shape and the U-shape, to the left, and the minimum distance between vehicles during the reconfiguration, total field approach and potential field approach compared, to the right.

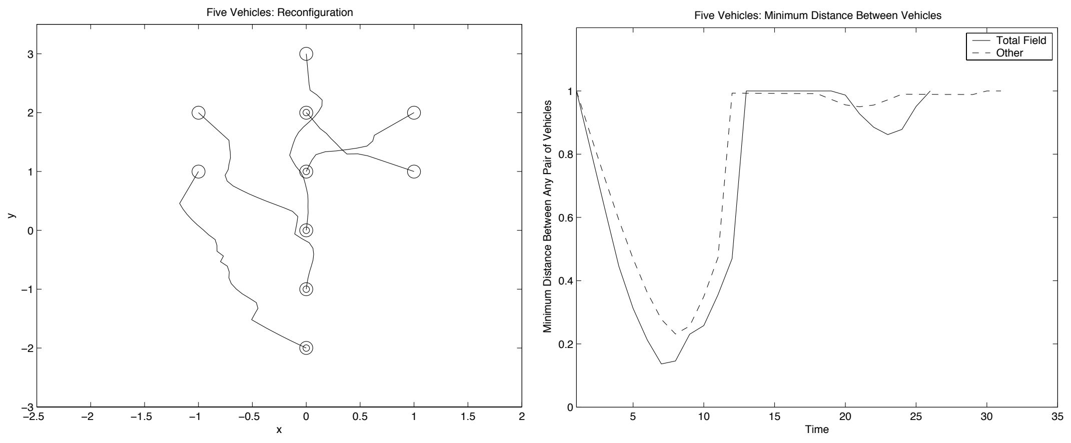

Figure 6.15: System reconfiguration between the I-shape and the U-shape, to the left, and the minimum distance between vehicles during the reconfiguration, total field approach and potential field approach compared, to the right.

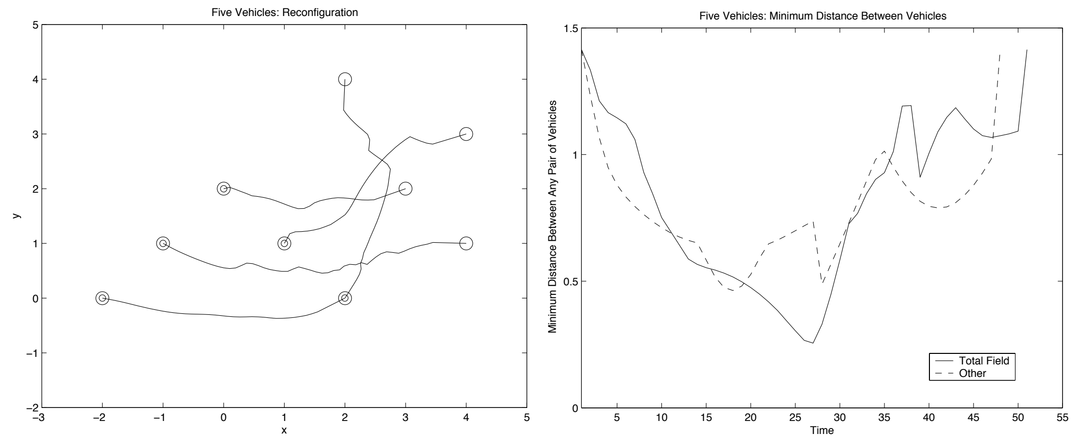

Figure 6.16: System reconfiguration between the V-shape and the M-shape, to the left, and the minimum distance between vehicles during the reconfiguration, total field approach and potential field approach compared, to the right.

Permutation Reconfiguration In figures (6.17), (6.18) and (6.19), we find simulations of permutation reconfigurations with performance comparisons betwen the total field approach and a standard potential field approach.

In figure (6.17), we see a permutation reconfiguration within the I-configuration, where the original node order $d_{original}$ is permuted into d_1,

$$d_{original} = \begin{Bmatrix} 1 & 2 & 3 & 4 & 5 \end{Bmatrix}, d_1 = \begin{Bmatrix} 3 & 4 & 5 & 1 & 2 \end{Bmatrix} \tag{6.32}$$

To the left in the figure, we see the reconfiguration trajectories, while to the right the minimum distance between vehicles during reconfiguration is shown as a function of time with the total field approach and a standard potential field approach compared. Without collision avoidance, this is a one-dimensional problem which will lead to several collisions; with the total field approach, the units react to the motion of the other vehicles and the collisions are avoided, as shown in solid line.

Figure (6.18) shows a permutation reconfiguration within the M-shape, where the original node order $d_{original}$ is permuted into d_2,

$$d_2 = \begin{Bmatrix} 2 & 3 & 1 & 5 & 4 \end{Bmatrix} \tag{6.33}$$

The symmetry of the subproblem for nodes 4 and 5 would lead to a collision with the naive scheme; the collision is avoided with the total field approach.

Figure (6.19), finally, shows a permuation reconfiguration within the V-shape; again, the node order $d_{original}$ is permuted into d_2 as given above.

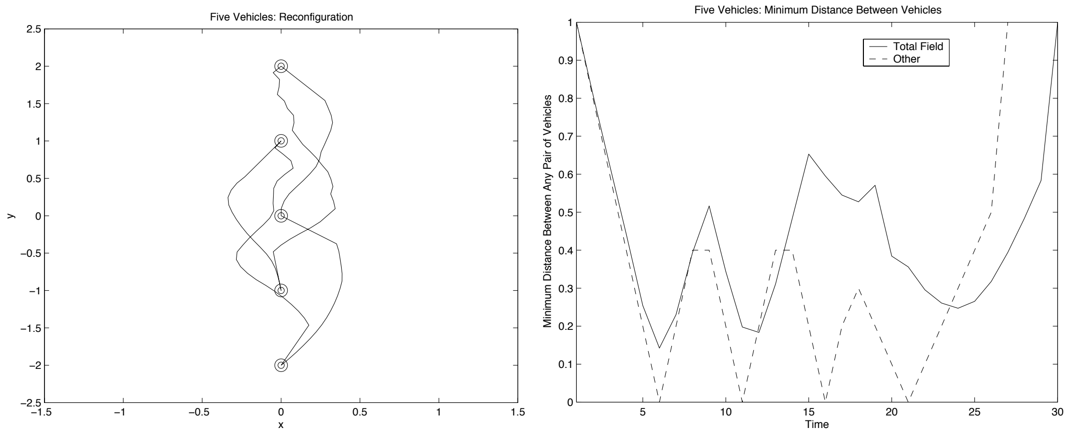

Figure 6.17: Permutation reconfiguration while keeping the I-shape, to the left, minimum distance between vehicles during the reconfiguration, total field approach and potential field approach compared, to the right.

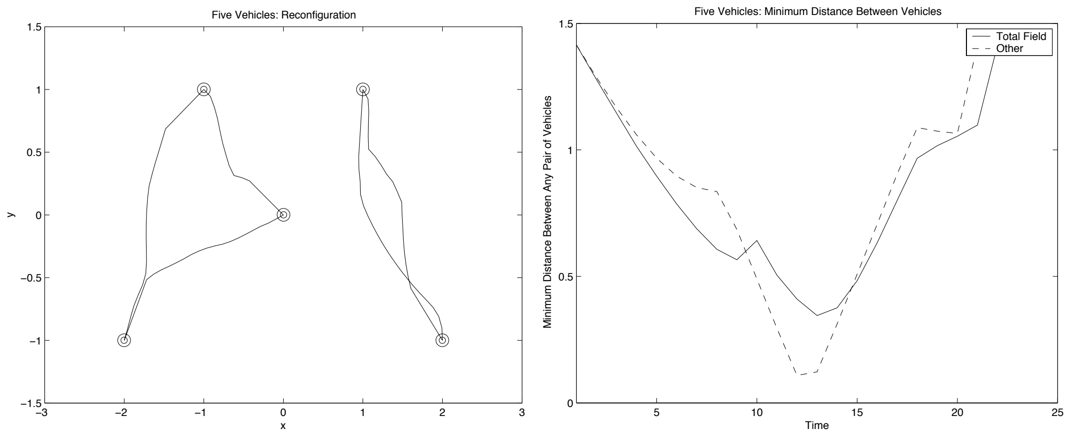

Figure 6.18: Permutation reconfiguration while keeping the M-shape, to the left, minimum distance between vehicles during the reconfiguration, total field approach and potential field approach compared, to the right.

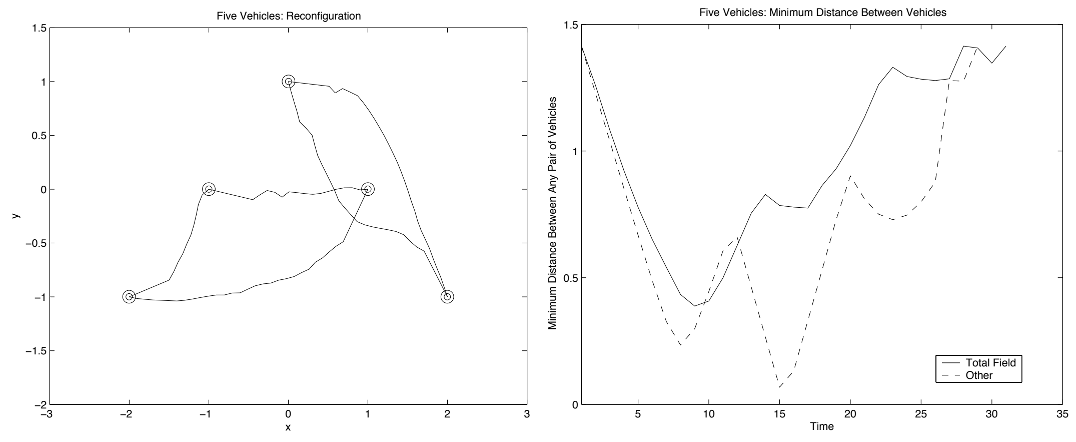

Figure 6.19: Permutation reconfiguration while keeping the V-shape, to the left, minimum distance between vehicles during the reconfiguration, total field approach and potential field approach compared, to the right.

Structure Reconfiguration In figure (6.20), (6.21) and (6.22), we see structure reconfigurations consisting in node additions after some nodes are lost and have to be replaced.

Figure (6.20) assumes an initial situation where nodes 2 and 4 of the I-shape have been lost and vanished and two new nodes are sent out to replace them. At the same time, a permutation reconfiguration takes place so that the original node order $d_{original}$ is replaced by d_3,

$$d_3 = \{2 \quad - \quad 3 \quad - \quad 4\} \tag{6.34}$$

Here, the dash denotes a lost node; the new nodes take roles 1 and 5. As seen to the right in the figure, neither the total field approach nor the naive approach with no collision avoidance leads to collisions but the margin is somewhat broader for the total field approach.

In figure (6.21), nodes 2 and 3 in the V-shape stop functioning but do not vanish; instead, they now become static obstacles that both the remaining and the new nodes have to avoid. A permutation reconfiguration occurs simultaneously so that the initial node order $d_{original}$ becomes d_4, while again the new nodes assume roles 1 and 5. In this case, the naive approach gives rise to a collision between a new node and the static former node 3; with the total field approach, the new node instead navigates between the two static nodes.

$$d_4 = \{2 \quad - \quad - \quad 3 \quad 4\} \tag{6.35}$$

Figure (6.22), finally, shows a structure reconfiguration of the M-shape where three new nodes are addded after nodes 3, 4 and 5 stop functioning and become obstacles. In this case, no simultaneous permutation reconfiguration takes place - the old nodes keep their previous roles. With the naive approach, one of the new nodes collides with the static former node 4, but the collision is avoided with the total field approach.

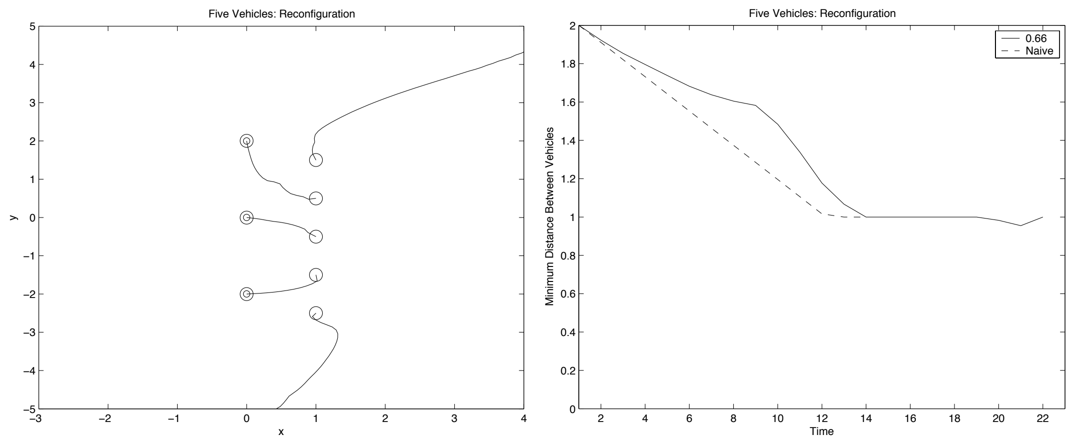

Figure 6.20: Structure reconfiguration where two nodes vanish from the I-shape and are replaced by two new nodes, to the left, minimum distance between vehicles during the reconfiguration with total field approach and naive scheme without collision avoidance compared, to the right.

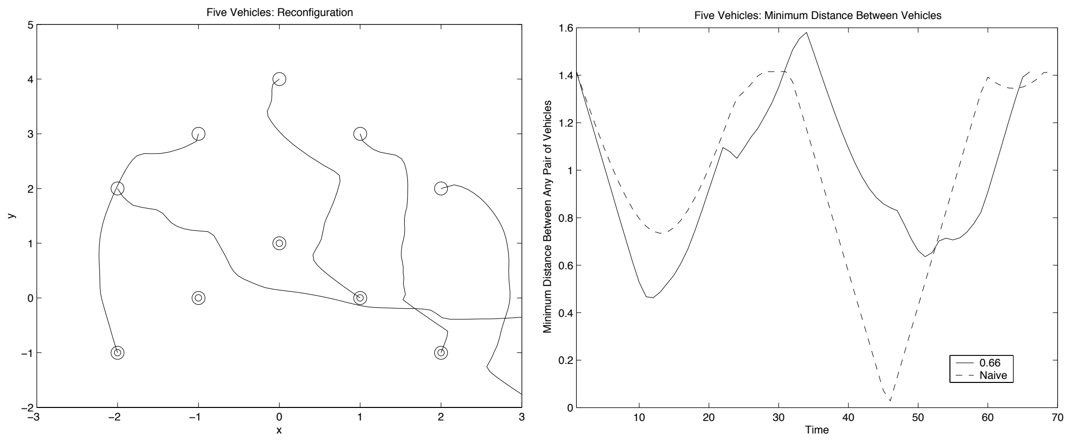

Figure 6.21: Structure reconfiguration where two nodes stop within the V-shape and are replaced by two new nodes that have to navigate past them, to the left, minimum distance between vehicles during the reconfiguration with total field approach and naive scheme without collision avoidance compared, to the right.

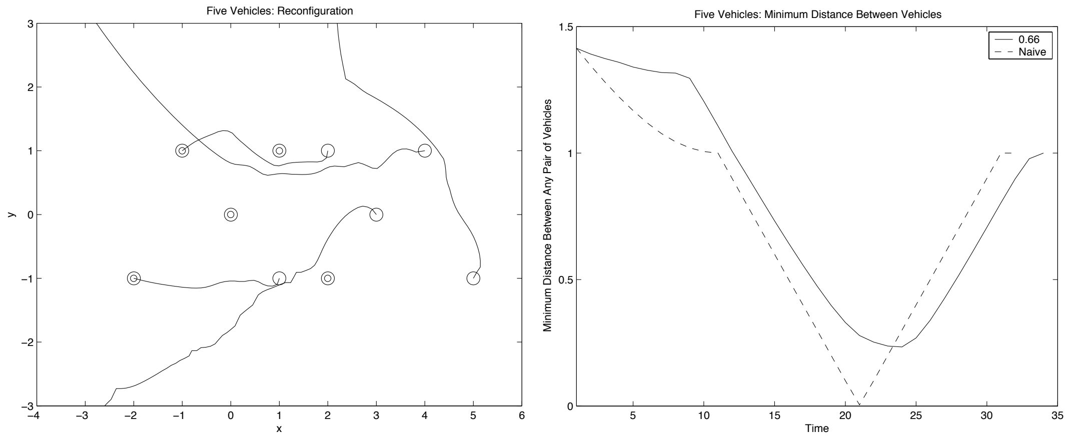

Figure 6.22: Structure reconfiguration where three nodes stop within the M-shape and are replaced by three new nodes that have to navigate past them, to the left, minimum distance between vehicles during the reconfiguration with total field approach and naive scheme without collision avoidance compared, to the right.

6.11 Hardware Implementation

We also performed experiments with hardware using one sensor and two identical light-weight magnets to simulate the scenario of two vehicles moving relative to each other. Furthermore, a series of measurements was carried out to confirm the theoretical deductions such as the strategic sensor positions and detection ranges and to test the function of the patented offset straps of the sensor - for the latter measurement series, we used two sensors and a set of magnets of varying sizes.

Sensors The sensors used were Honeywell's HMC2003 and HMR2300, which are three-axis magnetic sensors targeted at fields up to 2 gauss and have a resolution of less than 70 μgauss. The sensors are equipped with patented offset straps that allow the application of an electrically generated magnetic field in a direction opposite to the exterior magnetic field, thus making it possible to locally cancel out any component of the exterior field and bring the net field back to the linear range.

Magnets The magnets used to simulate two navigating vehicles were two identical, cylindrical magnets of length 18 mm, diameter 5 mm and weight 6 g - their magnetic moments were aligned with the cylinder axes.

Verifying Strategic Positions In the measurements to confirm the theoretical deductions, we verified the ratio between the detection range and the on-board distance between the vehicle's own sensors and magnet to be around thirty. This was also obtained theoretically by defining the on-board distance between the vehicle's own sensors and magnet as the distance where the sensors would measure 2 gauss and the detection range as the distance where the sensing vehicle would measure less than 70μgauss. We also verified the strategic sensor positions to correspond to those calculated above. The detection ranges for the different magnets tested were found to vary from 1m to 7m.

Testing Relative Motion In our simulation of a two-vehicle scenario, we positioned the sensor at a strategic position in Θ_x or Θ_y close to one of the magnets, thus simulating a vehicle with an on-board magnet and one sensor. The sensor and the first magnet were kept static throughout the test. The other magnet was mobile and used to simulate a second vehicle moving relative to the first one. The same sets of tests were performed once for Θ_x and once for Θ_y. We let the mobile vehicle travel along the x-axis and y-axis, respectively, first approaching and then passing the static vehicle. In this scenario, we got interference from the on-board magnet, situated as close as 1-2 cm to the sensor - this reduced the detection range ratio from thirty to fifteen, likely because the sensors used - HMC1002 as part of HMR2300 - do not compensate for cross-axis effects, as do more advanced sensors.

Chapter 7

Robustness

What happens when some units do not keep their individual positions in the current configuration? To what extent can configurations be kept and reconfigurations performed correctly in the presence of different kinds of such deviations?

In this section, we first review the notion of robustness in control and see how it is defined in central and distributed control, respectively. We then focus on what aspects of robustness are particularly relevant in the specific case of reconfiguration control and point at other control areas where analogous robustness issues are encountered. Based on these observations, we finally propose a definition of robustness for reconfiguration control and give a motivation.

7.1 Robustness in Control: Background

The notion of robustness in control refers to the insensitivity of the system behavior to deviations from the assumed model [23]. Examples of such deviations are parameter errors, noise distributions other than those assumed and unmodelled dynamics, such as nonlinear terms ignored in a linearization. Stability, convergence and optimality are especially important aspects of the system behavior in this respect.

A preciser definition of robustness is as follows: given a set $\bar{M}$ of possible models of the system and the observation noise and given a metric d_M over $\bar{M}$, we assume a model $M_0 \in \bar{M}$ is chosen based upon which a controller T is obtained. The controller T is defined as robust if it gives a sufficiently good performance not only at the model, but also in a neighborhood $\{M \in \bar{M}|\ d_M(M, M_0) < \epsilon\}$ of the model, where $\epsilon > 0$.

To understand the development of the notion of robustness in control theory we may need to briefly consider the history of control theory itself, starting by central control and continuing with distributed control theory.

7.1.1 Centralized Control

The first rigorous analysis of a feedback control system was given in 1868 by Maxwell, who linearized a set of differential equations of motion to find the characteristic equation of the system and showed how the system stability depends on the system parameters, thus explicitly stating the robustness of the system stability to errors in the system parameters.

In the development of mass communication systems at Bell Laboratories during the nineteen twenties and thirties, the frequency domain approaches proposed over a century earlier by Laplace and Fourier proved useful. This led Nyquist to propose the Nyquist stability criterion in 1932 and Bode to suggest in 1940 the use of the notions of gain and phase margin in stability theory - all of these notions also give the robustness margins.

In 1946, Hall pointed at the risks of ignoring noise in control theory and showed how the frequency domain approach could be used to design stable systems that took the noise into account.

However, the frequency domain approach is best suited for linear time-invariant (LTI) systems and, furthermore, its graphical techniques such as Nyquist plots are best applied to single-input single-output (SISO) systems. The desire in the nineteen fifties and sixties to design advanced spacecraft of nonlinear dynamics and with multiple inputs and outputs (MIMO) led to a return of the focus to the time domain.

In 1960, Kalman introduced linear algebra and matrices which facilitated the treatment of MIMO systems. In the nineteen seventies, the frequency domain approach was extended to MIMO systems, giving rise to the notions of characteristic locus, diagonal dominance and the inverse Nyquist array.

The development of stability theory for MIMO systems from the nineteen seventies and on thus gives a theoretical framework for making rigorous statements about the stability and robustness of systems that are spatially distributed but share all information; not only should each subsystem be stable and robust in the traditional SISO sense, but the system as a whole, involving interactions between these subsystems, must also be stable and robust. However, the complexity of such systems grows exponentially with the number of distributed units - with the current computational capacity, very large systems are only tractable with the distributed control paradigm, the robustness of which will be discussed below.

Thus, the robustness tools of central control are much the same as the tools for assessing stability since these automatically give the allowed interval for robustness. These include notions such as gain and phase margin, usually expressed in open-loop properties, such as the open-loop system matrix, that can be measured.

7.1.2 Distributed Control

In distributed control, there are more than one controller and information is not generally shared. The control of power systems and air traffic control have hitherto been the main focus of this relatively young control area.

Assuming that the controllers know each other's action spaces but not each other's strategies, a conservative method is the worst-case scenario approach where each controller avoids such actions that might lead to an undesirable state. This approach assumes that the controller knows a good deal about the structure of the environment, including the other agents, and that this structure is rather static; it also requires defensive behavior to be acceptable.

7.2 Robustness in Reconfiguration Control

Reconfiguration control may be addressed with methods either from central control or distributed control, although the poor scaling of central methods will generally put severe constraints on the size of the network.

7.2.1 Reconfiguration Specific Constraints

The two major points in reconfiguration control is configuration keeping and switch of configurations.

Once a configuration has been formed, it must be kept until a signal is given to change configurations. Configuration keeping must be robust to some units fulfilling their particular role in the configuration imperfectly. Furthermore, a configuration must not break down due to the total failure of just one or two units. If indeed the constant occupation of a small number of configuration roles is vital to the configuration keeping, there must be redundancy mechanisms ensuring that another unit takes over any of those particular roles, should one of its current occupants fail.

The switch of configurations is, as stated above, initialized by some agreed upon signal. It is most important that configuration switches are not started by mistake and that if accurately initialized, they are performed swiftly and safely. Thus, a reconfiguration should be robust to contamination of the agreed upon signal by noise and to the failure of some units to detect the signal. Furthermore, as discussed above in the section on mixed initiative, the network must be robust to conflicting reconfiguration commands given by different nodes almost simultaneously. On the other hand, a dynamic environment may require the possibility to interrupt or modify an initialized reconfiguration. If so, a signal for doing so must be devised that distinguishes such an emergency change of plans from the situation where conflicting commands are given close in time by parts of the network that ignore each other's commands.

7.2.2 Analogies in Other Control Areas

There are other areas of control where the quality of a collective result depends on the accuracy of a number of distributed inputs. In estimation theory, a parameter estimate is formed from a number of noisy measurements - scalar or vector - containing some information; whether the measurements are sequential or simultaneous, they can be seen as distributed inputs. The estimator aims at extracting this distributed information as efficiently as possible to output a collective result - the estimate - that is as accurate as possible.

Robustness in estimation theory generally means robustness to deviations from the probability distribution used to model the noise. As a notion in estimation theory, the term robustness was coined by Box in the nineteen fifties and initially defined as insensitivity to changes in extraneous factors not under test. Hampel later added the aspect of stability to the definition of robustness in analogy with the robustness of mechanical structures. Another important insight was the fact that a robust estimator must be insensitive not only to a large number of small errors but also to a small number of large errors [24, 25].

The theory of robust estimation offers a number of quantitative measures that might find their analogies in reconfiguration control. One such quantitative measure is the breakdown point, indicating the smallest fraction of contaminated observations that can cause an estimator to become completely unreliable; an analogy in reconfiguration control would be how many nodes may fail without noticeably changing the configuration. Another important quantitative measure is the minimax variance of an estimator which corresponds to the general question of how to minimize the worst-case performance, given a set of possible noise sources and a function for evaluating the system performance.

7.2.3 Proposed Definition

We propose to define robustness in reconfiguration control as the absence of influence on the system performance by small errors in a large number of nodes or by large errors in a small number of nodes. Thus, the keeping of a configuration C, $C = \{\mathbf{x} \in X^N \mid \mathbf{f_C}(\mathbf{x}) \in F_C\}$ by a network of nodes $k = 1, ..., N$ positioned at times t at
$\mathbf{y}(t) = [y^1(t), y^2(t), ..., y^N(t)]^T$, $y^k(t) \in X$, is seen as robust if

$$|d_X(\mathbf{y}(t)) - d_X(\mathbf{x})| < \epsilon$$

for some system specific but uniform ϵ and a performance measure d_X. Likewise, the reconfiguration starting at time t_1 and ending at time t_2 from a configuration C into a configuration C',
$C' = \{\mathbf{x} \in X^N \mid \mathbf{f_{C'}}(\mathbf{x}) \in F_{C'}\}$ is seen as robust if

$$\begin{aligned} |d_X(\mathbf{y}(t)) - d_X(\mathbf{x})| &< \epsilon \text{ for } t \leq t_1, \\ |d_X(\mathbf{y}(t)) - d_X(\mathbf{x})| &< \epsilon \text{ for } t \geq t_2 \end{aligned}$$

and $t_2 - t_1 \leq t_{max}$, with t_{max} being a system specific constant.

7.2.4 Motivation

The proposed definition conforms with the general definition of robustness in control and is, more specifically, analogous to the definition of robustness in estimation theory. The purpose of reconfiguration is adaptation to changes in the environment or in desired system functions in order to keep the system performance optimal or sub-optimal - the proposed definition reflects this by putting the system performance in focus.

How do the two implementations of our algorithm, Replicator learning and Cluster, meet the proposed requirements for robustness in reconfiguration control?

7.3 Robustness in Replicator Learning

As seen in the heterogeneous case presented above in the section on Replicator learning, the system is robust to at least some categories of large errors occuring in a large minority of the nodes. What categories of errors are relevant in the Replicator learning scenario and to what extent is the system robust to these possible errors?

7.3.1 Parameters: Matrix G

Parameter errors concern errors in the game matrix G used by each node and can be further divided into three types - the matrix may be contaminated by noise, the matrix may not correspond to any defined configuration or the matrix may correspond to a waypoint but not the current one; the last of these cases is treated as a reconfiguration error rather than a parameter error and will be discussed below.

A noisy matrix $\bar{G} = G + N$ used by some nodes would in general cause a local disturbance of those nodes which would, however, be compensated for by the other nodes like in the case of the heterogeneous nodes.

If the matrix used does not correspond to any defined configuration, it may have several ESS's or no ESS; in the former case, if all nodes used the same matrix, the system would converge to one of the ESS's, which one depending on the initial conditions. All finite matrices have at least one Nash equilibrium, many of which are centers in the replicator dynamics; thus, if all nodes used the latter matrix, an oscillating motion around the Nash equilibrium would be probable, as seen above in a simulation. If some nodes use the correct matrix and some nodes a different matrix, both populations will try to compensate for each other as the homogeneous nodes were seen to do for the heterogeneous ones above - the net result will depend on the proportion of the two types of nodes and on the distance between the current waypoint and the points corresponding to the multiple ESS's or the Nash equilibrium of the erroneous matrix; this is illustrated in figure (7.1).

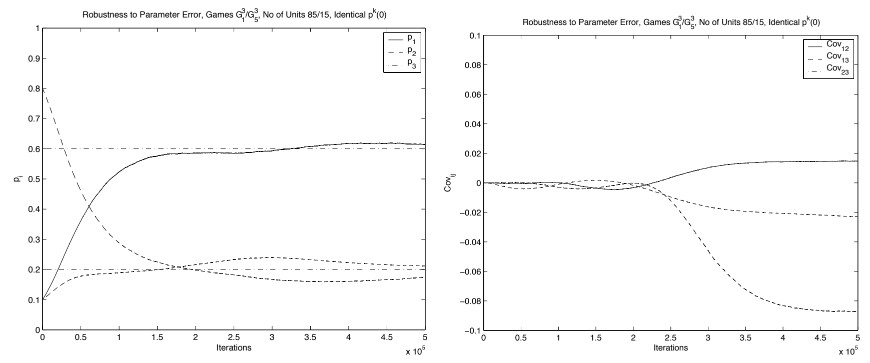

Figure 7.1: Imperfect convergence towards y_1^3 in the presence of nodes using matrix $\mathbf{G}_5^3$ instead of $\mathbf{G}_1^3$, to the left, corresponding covariances to the right; all units starting from the same point, N=100.

7.3.2 Information: Opponent Action

Information errors concern defects in the communication between a given node and the other nodes in the network. The node may not receive any information at all, it may receive some information or the information may be noisy and thus sometimes incorrect. A fourth type of information error would be an exterior attack where the node by mistake interacts with a node from another network rather than with a fellow node from its own network. Finally, the node may receive erroneous information - the last case may be seen as an interior attack where another node in the network sends out deceptive information on purpose about its actions.

If the node does not receive any information about the action of the other node although it was matched to another node, it may opt for the same approach suggested for the heterogeneous nodes, that is, generate a ficticious action from its own composition to replace the action taken by its opponent; this is illustrated in fiugre (7.2). With this approach, the node will converge towards the current waypoint but will not contribute towards forming a cluster around the waypoint.

Indeed, the option of generating a ficticious action from the current own composition can be used as a redundant safety measure even when information about the actions chosen by the opponent nodes is received; if the trajectories of the compositions generated with the regular algorithm diverges very much from that generated with the modified algorithm used by heterogeneous nodes, it is possible that the information obtained was incorrect, by mistake or intentionally.

If a node does not receive any information about the actions chosen by its opponents and has at the same time lost its parameters - thus, a combination of a parameter error and an information error - it would do best to move to the centroid of the region in which it is moving since that will require an equal effort for the other nodes to compensate for it no matter what the current waypoint is, given that the node has no information about the current waypoint.

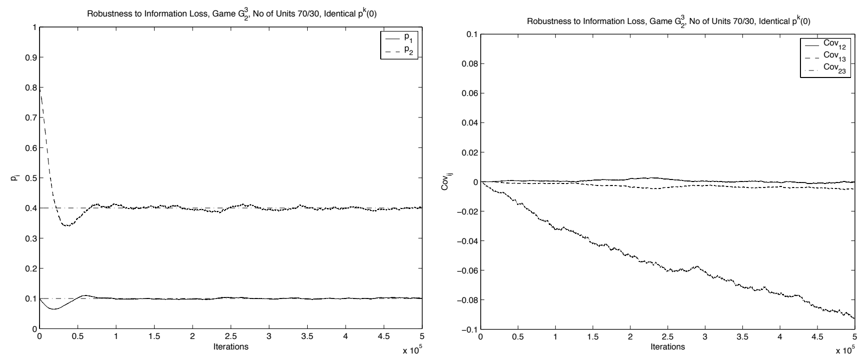

Figure 7.2: Convergence to y_2^3 to the left in the presence of information loss, corresponding covariances to the right; all units starting from the same point, N=100.

7.3.3 Operation

Operation errors concern the way in which the node implements the algorithm - the main case would be the static node which does nothing, yet is still part of the network. This slight variation of the heterogeneous case will be solved exactly as in the heterogenous case, since the static nodes can be seen as heterogeneous nodes starting out at an optimal operating point where they will thus stay.

7.3.4 Network Composition

Changes in network composition such as node losses or node additions may be intentional structure reconfigurations or may occur unintentionally and be seen as errors in network composition; in either case, they can be treated in the same way by the network. Thus, for this type of error, we can refer back to the section on structure reconfigurations where the network was found to be very robust to even quite large changes in network size.

7.3.5 Reconfiguration

Reconfiguration errors concern cases where the reconfiguration signal was not received by some nodes, where conflicting reconfiguration signals were given or where a deceptive reconfiguration signal was sent out either by a network node - an interior attack - or by an exterior node not part of the network - an exterior attack.

Redundancy of signals in the form of rules can be used as a precaution in case reconfiguration signals are lost - the nodes may have been informed of the preliminary order in which waypoints were to be visited before they were sent out. As suggested above, particularly dangerous waypoints may be visited only conditionally on visiting a particular waypoint immediately before.

To reduce the risk of interior nodes giving erroneous reconfiguration signals by mistake, a certain number of other nodes may be required to confirm the suggested reconfiguration before it is actually carried out. However, this will slow down the network and decrease its reconfiguration speed if too many nodes are involved. To prevent exterior nodes from giving reconfiguration signals, coded signals may be used.

7.4 Robustness in Cluster

The Cluster framework is in several ways different from the Replicator learning scenario, but the error categories listed above are still relevant in this framework too.

7.4.1 Parameters

The main parameters in which we are interested are the weight γ and the reference positions x_j^k used by vehicles $k = 1, ..., N$ for the configurations $j = 1, ..., j_{max}$.

Weight γ The relevant spectrum of γ, $0 < \gamma < 1$, was discussed above; for the purpose of guaranteeing arrival at the destination, we found the constraint $\gamma > 0.5$. If $\gamma \leq 0.5$, the vehicles soon exhibit an irrational behavior, overreacting to the presence of other vehicles - the resulting trajectories cross paths several times with the same vehicle and may head off in directions away from the goal.

As for the upper part of the spectrum for γ, we experimentally found it necessary to impose $\gamma < 0.7$ to avoid too narrow margins between vehicles; this thus leaves $0.5 < \gamma < 0.7$ as the interesting interval for γ.

From a robustness point of view, the system is more robust to some vehicles using too large a γ rather than a too small one; if a vehicle uses a γ that is too large, that vehicle will pay relatively less attention to collision avoidance than the others and leave it to the other vehicles to yield. However, if a vehicle uses a γ that is too small, it will generate a quite inconsistent trajectory and seriously disturb many other vehicles, potentially generating quite a few collisions.

Reference Position x_j^k The other interesting parameter is the reference position x_j^k each vehicle k seeks to keep with respect to the system mass center within the current configuration j. If a vehicle uses the wrong reference position $\hat{x}_j^k$, the mass center may not follow the expected trajectory but a trajectory consistently perturbed by $O(\frac{1}{N}|x_j^k - \hat{x}_j^k|)$. If the perturbed mass center trajectory can be compared to the expected trajectory, it may thus be possible to extract some information about which node is responsible, given that the correct reference positions of all nodes are known.

The conclusion is that the system is to some extent robust to this type of errors since the other nodes adapt to the perturbed mass center and that, if the number of nodes using defective reference parameters is not too large, it may even be possible to point at the defective node by analyzing the perturbed trajectory of the system mass center.

7.4.2 Information: Sensors

The vehicles obtain information about the other vehicles by sensing the total field with their magnetic sensors - what happens if some sensors stop working?

Partial Sensor Failure Four sensors are used to measure the field difference in the x-direction in the vehicle's reference frame; another four sensors measure the field difference in the y-direction. If some of the sensors are not working, a different formula may be used to calculate a simplified estimate of the gradient which, although less accurate, still carries information.

There is an asymmetry in this case for the sensing along the x-axis and the y-axis, respectively, since there are two sensor pairs along each axis for the sensing along the y-axis but only one sensor pair for each axis for sensing along the x-axis.

Both groups of four sensors are completely robust to sensor failure in one single sensor and are robust to respectively four or two out of the six possible combinations of simultaneous failure in two sensors - in the two remaining cases, no estimates will be obtained for two of the four components $\frac{\partial B_x}{\partial x}$, $\frac{\partial B_x}{\partial y}$, $\frac{\partial B_y}{\partial x}$ and $\frac{\partial B_y}{\partial y}$.

In figure (7.3), we see how the minimum distance between any pair of vehicles over time would change in two of the reconfiguration scenarios studied above if all vehicles lost the same one or two sensors. To the left in the figure, we again simulated the permutation reconfiguration shown in figure (6.18), this time removing either the front sensor x_1 or the back sensor x_3 from all vehicles. The partial derivative $\frac{\partial B_x}{\partial y}$ was now approximated as

$$\frac{\partial B_x}{\partial y} \approx \frac{x_1 - \frac{x_2+x_3}{2}}{\frac{1}{2} l_{x_{13}}} \text{ or } \frac{\partial B_x}{\partial y} \approx \frac{\frac{x_2+x_3}{2} - x_3}{\frac{1}{2} l_{x_{13}}}, \tag{7.2}$$

depending on what sensor input was missing, whereas B_x was approximated as

$$B_x \approx \frac{x_1 + x_2 + x_4}{3} \text{ or } B_x \approx \frac{x_2 + x_3 + x_4}{3}. \tag{7.3}$$

We see that although the simulated node loss slightly reduced the minimum distance between any pair of vehicles, the minimum distance trajectories were barely changed by the simulated loss of one node.

To the right in figure (7.3), we simulated a double node loss in the structure reconfiguration from figure (6.21). This time, all vehicles were assumed to simultaneously lose either the back sensor x_3 and the right back sensor y_2 or x_3 and the right front sensor y_1. The modified formulae for $\frac{\partial B_x}{\partial y}$ and B_x were still as given above; the partial derivatives $\frac{\partial B_y}{\partial x}$ and $\frac{\partial B_y}{\partial y}$ were now approximated as

$$\frac{\partial B_y}{\partial x} \approx \frac{y_1 - y_4}{l_{y_{14}}} \text{ and } \frac{\partial B_y}{\partial y} \approx \frac{y_4 - y_3}{l_{y_{43}}} \text{ or} \tag{7.4}$$

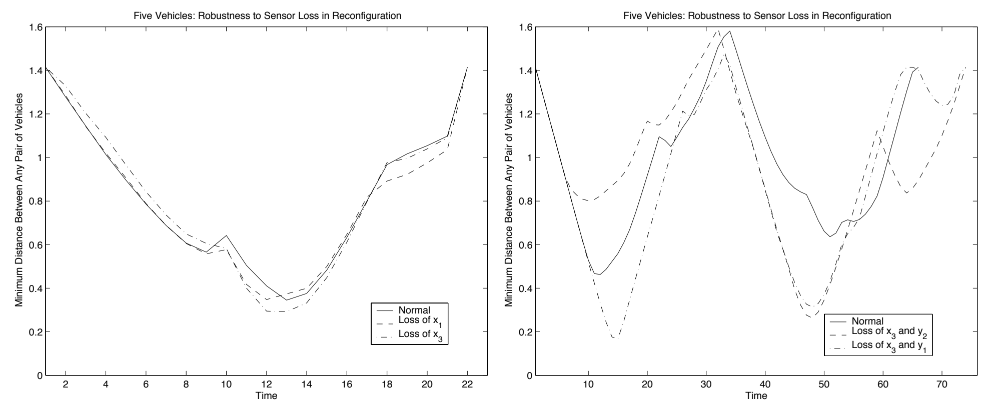

Figure 7.3: Minimum distance between any pair of vehicles during the permutation reconfiguration shown above in figure (6.18) with and without loss of one sensor, to the left; minimum distance between any pair of vehicles during the structure reconfiguration shown above in figure (6.21) with and without loss of two sensors, to the right.

$$\frac{\partial B_y}{\partial x} \approx \frac{y_2 - y_3}{l_{y_{23}}} \text{ and } \frac{\partial B_y}{\partial y} \approx \frac{y_4 - y_3}{l_{y_{43}}} \tag{7.5}$$

with B_y approximated as

$$B_y \approx \frac{y_1 + y_3 + y_4}{3} \text{ or } B_y \approx \frac{y_2 + y_3 + y_4}{3}. \tag{7.6}$$

We see that the change in the minimum distance trajectories were now much more noticeable, although the qualitative behavior was essentially the same. The minimum distance with a double node loss was now less than half of that obtained when all sensors were intact.

Complete Sensor Failure If all sensors fail, the vehicle completely relies on the other vehicles to yield to it - this corresponds to the situation described above where there is an asymmetry in roles and the vehicle without sensors is playing the role of the dominant vehicle. To reduce the risk of collisions, the impaired vehicle may start moving slower or stop altogether for a limited period to let the others pass.

7.4.3 Operation

Operation errors are errors in implementing the algorithm; they include incorrectly identifying the current formation, mistaking the goal or taking too large steps or steps of varying size.

Current Formation If one or a few nodes know the index of their current role but incorrectly identify the current formation, in the worst-case scenario they will try to occupate a role filled by another node, which will lead to a collision. Since the reconfiguration is normally a transient process of all nodes finding their respective relative positions, the fact that a node continues detecting a large field close to its goal is by itself an error signal which may cause both nodes to either leave the network or to request an information update.

Current Goal Likewise, a node that has incorrect information about the direction of the goal or the current location of the mass center will encounter an unexpectedly large or unexpectedly small total field which will serve as an error indication.

Low Precision A node which takes larger than expected steps can be reallocated to a front position where it will speed up the network and not cause collisions with other nodes. Low precision nodes that track their positions with respect to the system mass center less precisely than other nodes may be allocated to end or side positions where they are less likely to disturb the trajectories of other nodes than if they were given interior positions.

7.4.4 Network Composition

Just as in the Replicator learning case, structure reconfigurations such as additions or losses of nodes may be intentional or unintentional and may be treated in similar ways regardless of why they occurred. However, since this scenario puts much severer constraints on the individual node positions, the higher precision required makes it necessary to treat the unintentional node losses somewhat differently.

Addition or Loss of Nodes If a node is accidentally lost and this is noticed by the coordinator pushing the mass center forward along the desired trajectory, the diverging position of the lost node is not taken into account when calculating the location of the mass center; instead a dummy node is created which is always assumed to track the mass center perfectly until a replacement node has arrived.

If the node loss is not detected, the trajectory of the mass center will start to diverge too - as noted above, if one or only few nodes are lost, the direction of this divergence can be used to infer which node was lost.

Chapter 8

Applications

Although modular robotic systems with a decentralized control architecture have been studied since the late nineteen eighties, no one has yet succeeded in taking this emerging technology out of the laboratory. What are the main fields of application and the corresponding system functions desired? What key issues remain to be addressed to take the final step out of the laboratory?

8.1 Application Fields

Automotive Automated highway systems are an important application in which cars and trucks are provided with autopilotes tracking the vehicles in front of them during navigation on the highway.

Aeronautical/Military Fleets of unmanned aerial vehicles (UAVs) navigating in dynamic formations are an active field of research. During this decade, a significant portion of the military aviation is planned to be replaced by unmanned aircraft.

Robotics Assembly robots have been used for the last decades in the industry; several robots are usually involved but act in sequence rather than in parallel. Operation stops due to robot failures are very costly but not so uncommon - an added challenge to launching robots working in parallel would thus be the increased risk of system failure. Therefore, the introduction of robot systems would have to be motivated by a significantly higher output and a lower risk of failure in individual robots.

Science Important are also a number of scientific applications in which a large number of distributed sensor units circling Earth are used to collect data, thus providing stereo sensing.

8.2 Application Tasks

Data Collection Swarm systems are suitable for a range of data collection tasks, where a static or dynamic physical field such as Earth's magnetic field is to be studied. By spreading a sensor swarm over an area and using their sensed data, a composite stereo picture is obtained of the field studied. The possibility to reconfigure such systems would allow users to make the resolution higher in areas of particular interest and adapt to changes in such areas.

Detection Detection tasks involve elastic mobile fences formed by swarm members around ships and other vehicles travelling by land or sea; such tasks are referred to by the term perimeter defense whereas the individual robots in the fence are called robotic sentries. The sentries here aim at detecting mobile enemies and preventing the enemy from getting through the fence. This is an example of an interaction between two reconfigurable systems - by reconfiguring, the sentry system may divide into modules, some of which encircle detected enemies while others continue to look out for undetected ones.

Another type of the detection scenario is the search-and-find scenario where static objects such as landmines are to be detected - this often involves operation in dangerous or inaccessible environments where the risk of losing units is high and the possibility of permutation and structure reconfigurations therefore essential.

Tracking Tracking mobile objects such as other vehicles used as references or being pursued is yet another application - members of a mechatronic system may track each other, as is common in the leader-follower approach to formation control, and the mechatronic system may in turn collectively track a mobile solitary vehicle such as a ship in perimeter defense around which the collective reconfigures for optimal defense.

Manipulation/Displacement Solitary robots are already used in the automotive industry for car assembly; a coordinated robot team can be particularly useful in handling elastic material and to allow a fleet of small robots to collectively handle large objects. This may either mean simple collective carrying of a large object from one location to another or involve handling such as manipulation or assembly of objects.

Mission The army ant scenario is a type military mission scenario, where a set of small robots collectively locate an object to retrieve, assume an appropriate configuration in order to pick up and carry the object, and successfully bring it back to their base. As reflected in its name, this is an entire scenario involving a sequence of collective tasks belonging to the above categories.

8.3 Application Objects

Since most of the described application systems are intended to interact with objects or fields, collecting data, carrying and manipulating one or several objects or looking out for potential enemies, another relevant classification criterion concerns the very objects with which the system interacts, notably their number and their mobility.

Solitary or Collective? Does the system interact with one or several objects at a time? In tasks involving manipulation or displacement, the object is often a solitary large object which a node could not handle on its own but which collectively the nodes can manage to carry. The system may also handle one object at a time with all nodes performing different tasks simultaneously, such as painting different parts of a large object.

If instead the system is interacting with many objects at a time, we are closer to a scenario of a swarm-swarm interaction, which opens for a very large number of solutions.

Static or Mobile? A mechatronic system looking for hidden landmines in a given area can be seen as an interaction between a mobile swarm and a static swarm. An even more challenging scenario is when the mobile mechatronic system meets a similar mobile system, as may occur in perimeter defense or in encounters between groups of UAVs.

8.4 Out of the Laboratory

What steps remain in order to take the technology represented by distributed mechatronic systems out of the laboratory?

Safety A principal issue is safety in several respects - safety for persons who might get in the way, safety for equipment and buildings situated where the distributed system is operating and safety for the mechatronic units themselves.

Main principles for achieveing safety include separating the mechatronic system from anything vulnerable as far as possible and having a hierarchy of safe emergency solutions if something should go wrong.

Efficiency For distributed mechatronic systems to be the winning alternative in a choice between the novel technology and traditional, centralized control systems, the former alternative must be shown to be superior in efficiency.

Chapter 9

Conclusion

In this final chapter, we review what we have learned in reconfiguration control, point at the contributions of this thesis and suggest directions for future research in reconfiguration control.

9.1 Summary

The recent application of the distributed control paradigm to networks of mobile mechatronic units such as robots or unmanned vehicles offers powerful tools to handle large and complex systems that would be untractable with centralized control. At the same time, a set of new issues emerge, intrinsically associated with distributed control, such as cooperation, coordination, mutual adaptation and reconfiguration. Many of these central notions still lack generally agreed upon definitions and are interpreted in distinct ways in different settings and by different authors, although a general idea of a win-win situation or mutual coordination is usually present.

Reconfiguration Control

We found the topic of reconfiguration control, comprising configuration keeping and switching between configurations in distributed systems, to be a well delimited subject of study in distributed control that would permit us to address a number of interesting theoretical aspects of distributed control within a framework of practical interest in applications such as collective search and navigation in dynamic formations.

The topic of reconfiguration control has recently been addressed in various contexts such as formation control, multiagent learning, role allocation problems and swarm theory. However, most of these studies have focused on particular parts of reconfiguration control such as configuration keeping, and the solutions proposed have often been specifically targeted at the chosen application - we wanted to find a general framework that could fit reconfiguration problems from all the different fields cited above. To better understand the set of possible reconfiguration control problems, we analyzed and proposed classifications of configuration spaces and reconfigurations and identified coordination, adaptivity, mixed initiative and emergence as important features of a reconfiguring system.

While most previous studies focus on the subproblems of getting into configuration and keeping a configuration, we found that there is more to efficient reconfiguration control than just these aspects - indeed, the system must at all times be prepared for reconfiguration, which in the Replicator scenario meant that it was better if the units were spread out rather than concentrated around the ESS. This reflects the need to find a system balance between stability or configuration keeping on the one hand and adaptivity or reconfiguration on the other.

Our Approach

We proposed a novel theoretical analysis of reconfiguration control which interpreted node positions as strategies, identified each configuration with the unique equilibrium of a parametrized game and interpreted each reconfiguration as a switch of game parameters; an alternative way of changing the game was to add or reduce the information available.

Two Scenarios

Our proposed approach was implemented in two scenarios, the first one a swarm problem and the second one a traditional formation control problem; the scenarios were chosen to represent the two ends of a spectrum ranging from large-scale systems where the individual positions play a smaller role to small-scale system where high precision is required in the adjustment of the individual positions.

In the first scenario, Replicator learning, where the average position of a network of N units was required to visit a sequence of waypoints, we matched the desired average positions to the totally mixed ESS's of different games and saw the switch from one waypoint to another as a system reconfiguration. Permutation reconfigurations and structure reconfigurations were also described; furthermore, this simple network allowed us to address a number of issues such as how to handle heterogeneity, two-level polymorphism and robustness to factors such as parameter errors, loss of information and defective node operation.

In the second scenario, Cluster, we constructed a set of games which had the desired formations as their respective unique Nash equilibria and used total field collision avoidance for navigation during reconfiguration; in this way, the vehicles could avoid collisions without knowing the positions of any of the other vehicles. In this setting we also studied system, permutation and structure reconfigurations and addressed robustness issues, particularly concerning parameter errors and defective operation in nodes.

9.2 Contributions

We proposed a unified game theoretic framework for reconfiguration control that interpreted node positions as strategies, identified each configuration with the unique equilibrium of a particular game and saw reconfigurations as switches between games. A key part of our approach consisted in identifying classes of games having a desired equilibrium pattern and in using compact and uniform game descriptions. Scalability, adaptivity, generality and robustness were properties that motivated a game theoretic approach. We also showed how the framework could be adapted to fit heterogeneous nodes by redefining the payoff function for the heterogeneous nodes.

We introduced Replicator learning as a generalization of replicator dynamics, derived its equations and showed that for each totally mixed ESS q, there is a matrix G such that q is the unique ESS of G and the global attractor in replicator learning.

The Replicator learning scenario was linked to the application of collective search where one not only wants the average position of the units to coincide with a given point but also wants the units to spread out to increase their collective observation field. In our model, this corresponded to a desire to have a μ_∞ that was diffused rather than concentrated in a Dirac measure.

An additional advantage of the algorithm was its rapid convergence rate and the fact that given a desired ESS, a game matrix G could easily be constructed by solving a linear underconstrained system of equations.

Another aspect of the Replicator learning scenario concerned giving a control theoretic interpretation of the game theoretic framework, thus obtaining a framework of high-level control where each individual unit calculated its individual control as a function of a simple estimate; the individual trajectories were controlled just enough for the system to reach its collective goal.

The scenario was found to be very robust to heterogeneity, parameter error, loss of information and defective operation in individual nodes. In particular, the high robustness of the Replicator learning scenario to structure reconfigurations led to interesting analogies with biological systems such as the human liver that can survive losses of up to 80% of the nodes and the brain which can to a varying extent redistribute local network functions from damaged tissue to other areas.

The Cluster framework offered a game theoretic interpretation of a tracking problem extensively studied in formation control, thus showing that our unified framework was applicable also to a scenario where the individual positions required much more precise control.

To make the vehicles navigate safely in reconfiguration without knowing each other's positions, we introduced a novel collision avoidance algorithm in the form of a total field approach of magnetic nature. By strategic positioning of the magnetic sensors orthogonal to the vehicle's own field, we were able to naturally eliminate the vehicle's own field which otherwise would have drowned the field generated by the other vehicles, situated much farther away from the sensor than the vehicle's own magnet. We tested the principle behind the strategic sensing using two different sensors.

Finally, we proposed a definition of robustness in reconfiguration control based on an analogy with estimation theory and illustrated with examples how it would apply in our case.

9.3 Adaptivity vs. Stability

Reconfiguration control is more than just a successive convergence to a set of different equilibria; efficient reconfiguration implies that the network is at all times ready for change. This, in turn, requires a balance between adaptivity and stability. On the one hand, configuration keeping should be stable and robust, on the other hand, given the right conditions, reconfigurations should be swift and robust.

In the Replicator learning scenario, the additional term we derived prepares the system for reconfiguration by spreading the units away from the ESS, whereas the original term achieves convergence towards the equilibrium.

A general proposition concerned letting particularly aggressive configurations be conditional configurations - this would apply in particular for the Cluster scenario where the individual units have to keep precise positions and are not at liberty to modify their positions to prepare for a possible reconfiguration.

9.4 Future Work

Future work in this field includes the introduction of more communication between nodes and the study of interactions between reconfiguring systems.

More information exchange makes it possible to further pursue the idea of changing the game by adding or reducing information rather than changing the game parameters.

Another scenario concerns the play between polymorphism at the global level and polymorphism at the individual level, which is already introduced here since each individual player can as a rule play any strategy $p \in \mathcal{P}(A)$. To limit the polymorphism at the individual level, one could associate each player with a shifting pure strategy, for example decided as $m^k = \arg\max_i p_i^k(t)$.

For replicator learning, it would also be of great interest to study the dynamics of the higher moments of p, in particular the variance.

The extent of robustness to structure reconfigurations is another relevant area, implying either a swift permutation reconfiguration where remaining nodes fill the positions of the most essential nodes lost or a thoroughly parallel system structure where each subpart of the system is essentially the same. The former case was illustrated by our simulations where all nodes started from the same points whereas the latter case corresponds to the simulations where the units started from positions chosen uniformly at random all over the simplex. As noted above, this also has biological analogies in systems such as the liver and the brain.

The modularity of a distributed system is also an interesting field, allowing the system to temporarily divide itself in two or more separated subsystems that can then merge again. This can be achieved by letting one half of the nodes use one game matrix and the other half another matrix; to compensate for the nodes playing a different game, the nodes will become even further polarized.

If the division into subsystems is permanent rather than temporary, we instead obtain two or more separate systems and come to study the interaction between reconfiguring systems. In a competitive or aggressive situation, this would add an anti-cooperative element to the cooperative framework proposed - thus, even more attention has to be given to reconfiguration initiation to make sure that exterior nodes cannot pose as nodes part of the network and deceptively initiate reconfigurations in the other network.

Thus, reconfiguration control is a rich field with applications ranging from mechatronic systems to biological networks and offers a great many interesting problems to address!

Bibliography

[1] Robert Axelrod and William D. Hamilton. The evolution of cooperation. *Science*, 211:1390–1396, March 1981.

[2] Albert-Laszlo Barabasi and Albert Reka. Emergence of scaling in random networks. *Science*, 286:509–512, October 1999.

[3] Randal W. Beard, Jonathan Lawton, and Fred Y. Hadaegh. A coordination architecture for spacecraft formation control. *IEEE Transactions on Control Systems Technology*, 9(6):777–790, November 2001.

[4] Gerardo Beni. The concept of cellular robotic systems. In *Proceedings of of the Third International IEEE Symposium on Intelligent Control*, pages 57–62, Arlington, VA, August 1988.

[5] Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming.* Athena Scientific, 1996.

[6] D.T. Bishop and C. Cannings. A generalized war of attrition. *Journal of Theoretical Biology*, 70:85–124, 1978.

[7] Eric Bonabeau, Marc Dorigo, and Guy Théraulaz. *Swarm Intelligence: From Natural to Artificial Systems.* Oxford University Press, 1999.

[8] Tilman Börgers and Rajiv Sarin. Learning through reinforcement and replicator dynamics. *Journal of Economic Theory*, 77:1–14, 1997.

[9] Alfred M. Brückstein. Why the ant trails look so straight and nice. *The Mathematical Intelligencer*, 15(2):59–62, 1993.

[10] C. Cannings and G.T. Vickers. Patterns of ESS's II. *Journal of Theoretical Biology*, 132:409–420, 1988.

[11] David K. Cheng. *Field and Wave Electromagnetics.* Addison-Wesley, 2nd edition, 1989.

[12] David F. Chichka. Satellite clusters with constant apparent distribution. *Journal of Guidance, Control and Dynamics*, 24(1):117–122, January-February 2001.

[13] D.F. Chichka, J.L. Speyer, and C.G. Park. Peak-seeking control with application to formation flight. In *Proceedings of the 38th IEEE Conference on Decision and Control*, December 1999.

[14] Robert H. Crites and Andrew G. Barto. Elevator group control using multiple reinforcement learning agents. *Machine Learning*, (33):235–262, 1999.

[15] Eric van Damme. *Stability and Perfection of Nash Equilibria*. Springer-Verlag, 1987.

[16] Raffaello D'Andrea. Robot soccer: A platform for systems engineering. *Computers in Education Journal*, 10(1):57–61, 2000.

[17] Dipankar Dasgupta. Artificial neural networks and artificial immune systems: Similarities and differences. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, Orlando, FL, October 1997. IEEE.

[18] D.T. Edmonds. *Electricity and Magnetism in Biological Systems*. Oxford University Press, 2001.

[19] Michael D. Feezor, F. Yates Sorrell, and James G. Bellingham. Autonomous underwater vehicle homing/docking via electromagnetic guidance. *IEEE Journal of Oceanic Engineering*, 26(4):515–521, 2001.

[20] Veysel Gazi and Kevin M. Passino. Stability analysis of swarms. In *Proceedings of the American Control Conference*, pages 1813–1818, Anchorage, AK, May 2002.

[21] Veysel Gazi and Kevin M. Passino. Stability analysis of swarms in an environment woth an attractant/repellent profile. In *Proceedings of the American Control Conference*, pages 1819–1824, Anchorage, AK, May 2002.

[22] Brian P. Gerkey and Maja J. Matarić. Sold!: Auction methods for multi-robot coordination. *IEEE Transactions on Robotics and Automation, Special Issue on Multi-robot Systems*, 18(5):758–768, October 2002.

[23] Torkel Glad and Lennart Ljung. *Reglerteori: Flervariabla och olinjära metoder*. Studentlitteratur, 1997.

[24] Frank R. Hampel. Robust estimation: A condensed partial survey. *Z. Wahrscheinlichkeitstheorie verw. Geb.*, (27):87–104, 1973.

[25] Peter J. Huber. *Robust Statistics*. Wiley, 1981.

[26] Vikram Kapila, Andrew G. Sparks, James M. Buffington, and Qiguo Yan. Spacecraft formation flying: Dynamics and control. *AIAA Journal of Guidance, Control and Dynamics*, 23:561–564, 2000.

[27] Oussama Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, 5(1):90–98, 1986.

[28] Y. Koren and J. Borenstein. Potential field methods and their inherent limitations for mobile robot navigation. In *IEEE Conference on Robotics and Automation*, pages 1398–1404, Sacramento, CA, April 1991.

[29] H.J. Kushner. *Approximation and Weak Convergence Methods for Random Processes.* M.I.T. Press, Cambridge, MA, 1984.

[30] Serge Lang. *Analysis II.* Addison-Wesley Publishing Company, Inc, 1969.

[31] Naomi Ehrich Leonard and Edward Fiorelli. Virtual leaders, artificial potentials and coordinated control of groups. In *Proceedings of the 40th IEEE Conference on Decision and Control*, pages 2968–2973, 2001.

[32] Michael L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 157–163, San Francisco, CA, 1994.

[33] John Maynard Smith. The theory of games and the evolution of animal conflicts. *Journal of Theoretical Biology*, 47:209–221, 1974.

[34] Malcolm McCaig. *Permanent Magnets in Theory and Practice.* John Wiley & Sons, 1977.

[35] Colin R. McInnes. Autonomous ring formation for a planar constellation of satellites. *Journal of Guidance, Control and Dynamics*, 18(5):1215–1217, 1995.

[36] Colin R. McInnes. Simple analytic model of the long-term evolution of nanosatellite constellations. *Journal of Guidance, Control and Dynamics*, 23(2):332–338, March-April 2000.

[37] Roger B. Myerson. *Game Theory: Analysis of Conflict.* Harvard University Press, 1991.

[38] John Nash. Non-cooperative games. *The Annals of Mathematics*, 54:286–295, 1951.

[39] Reza Olfati-Saber. Flocking with obstacle avoidance. Technical report, Caltech, 2003.

[40] Eric Rasmusen. *Games and Information.* Blackwell, 2nd edition, 1994.

[41] Craig W. Reynolds. Flocks, herds and schools: a distributed behavioral model. *Computer Graphics*, 21(4):25–34, 1987.

[42] Tuomas W. Sandholm and Robert H. Crites. Multiagent reinforcement learning in the iterated prisoner's dilemma. *BioSystems*, 1995.

[43] Shankar Sastry. *Nonlinear Systems: Analysis, Stability and Control.* Springer-Verlag, 1999.

[44] Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *ICML*, pages 330–337, 1993.

[45] Peter D. Taylor and Leo B. Jonker. Evolutionarily stable strategies and game dynamics. *Mathematical Biosciences*, (40):145–156, 1978.

[46] Yuri Ulybyshev. Long-term formation keeping of satellite constellation using linear-quadratic controller. *Journal of Guidance, Control and Dynamics*, 21(1):109–114, January-February 1998.

[47] H. Van Dyke Parunak, Michael Purcell, and Robert O'Connell. Digital pheromones for autonomous coordination of swarming uav's. In *Proceedings of AIAA 2002*, 2002.

[48] G.T. Vickers and C. Cannings. Patterns of ESS's I. *Journal of Theoretical Biology*, 132:387–408, 1988.

[49] Tamás Vicsek, András Czirók, Eshel Ben-Jacob, Inon Cohen, and Ofer Shochet. Novel type of phase transition in a system of self-driven particles. *Physical Review Letters*, 75(6):1226–1229, 1995.

[50] P.K.C. Wang, F.Y. Hadaegh, and K. Lau. Synchronized formation rotation and attitude control of multiple free-flying spacecraft. *Journal of Guidance, Control and Dynamics*, 22(1):28–35, January-February 1999.

[51] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, June 1998.

[52] Jörgen Weibull. *Evolutionary Game Theory.* The MIT Press, 1995.

[53] Norbert Wiener. *Cybernetics, or control and communication in the animal and the machine.* John Wiley and Sons, Cambridge, Massachusetts, 1948.

[54] Hans S. Witsenhausen. Separation of estimation and control for discrete time systems. *Proceedings of the IEEE*, 59(11):1557–1566, November 1971.