

# PRELIMINARY STUDIES OF DYNAMICS OF PHYSICAL AGENT ECOSYSTEMS

**Thesis Proposal**

**Author: Israel Muñoz Moreno**

**University of Girona**

[imunoz@eia.udg.es](mailto:imunoz@eia.udg.es)

**July 2000**

# INDEX

INDEX .....	0
I. PREFACE.....	3
1.1 INTRODUCTION .....	3
1.2 INTRODUCTION TO RESEARCH WORK .....	3
1.3 ORGANISATION OF THIS WORK .....	4
II. STATE OF THE ART.....	5
2.1 PHYSICAL AGENTS.....	5
2.2 SELF-ORGANISATION AND EMERGENT BEHAVIOURS .....	6
2.3 ECOSYSTEMS .....	8
2.4 HETEROGENEITY IN MULTI-AGENT SYSTEMS.....	8
2.5 CONSENSUS.....	11
2.6 FOOTBALL DOMAIN.....	11
III. GUIDELINES OF THIS WORK.....	13
IV IMPLEMENTATION.....	15
4.1 INTRODUCTION .....	15
4.2 JAVASOCCER .....	15
4.3 BENCHMARK GENERATOR: IMPLICIT OPPONENT.....	17
4.3.1 Introduction.....	17
4.3.2 Formulation of the Benchmarks Generator .....	18
4.3.3 Purpose .....	18
4.3.4 Description of Parameters .....	18
4.3.5 The Time to Do the Experiment.....	19
4.3.5 The Features of the Ball.....	19
4.3.6 The Constraints .....	20
4.3.7 The Performance Index .....	20
4.3.8 Summary.....	20
4.4 ECOSYSTEM FORMULATION .....	21
4.4.1 Introduction.....	21
4.4.2 Choice equation .....	24
4.4.3 Agent Performance.....	25
4.4.4 Ecosystem and co-operative perception.....	28
4.4.5 Creating diversity.....	29
4.6 ROLES .....	31
4.7 CONSENSUS TECHNIQUE.....	32
4.8 IMPLEMENTATION.....	33
V EXPERIMENTS .....	35
5.1 INTRODUCTION .....	35
5.2 EXPERIMENTAL CONDITIONS .....	35
5.3 BENCHMARK DEFINITION.....	35
5.4 EXPERIMENTS WITH 1 PLAYER .....	37
5.4.1 Inclination 1 degree .....	37
5.4.2 Inclination 3 degrees.....	38
5.4.3 Inclination 7 degrees.....	40
5.5 EXPERIMENTS WITH 5 PLAYERS.....	41

5.6 CHANGES IN DYNAMICS .....	46
<b>VI CONCLUSIONS .....</b>	<b>48</b>
<b>VII FUTURE WORK .....</b>	<b>49</b>
7.1 TECHNICAL .....	49
7.1.1 <i>Role Design</i> .....	49
7.1.2 <i>New Knowledge Resources</i> .....	49
7.2 RESEARCH.....	49
7.2.1 <i>Reward Assignment</i> .....	49
7.2.2 <i>Equations</i> .....	50
7.2.3 <i>Diversity Analysis</i> .....	51
7.2.4 <i>Teams Performance Analysis and Optimal Solution</i> .....	51
<b>VIII BIBLIOGRAPHY AND REFERENCES .....</b>	<b>52</b>
8.1 MAIN BIBLIOGRAPHY .....	52
8.2 COMPLEMENTARY BIBLIOGRAPHY .....	56

**ANNEXE I:** Football team implementation

**ANNEXE II:** New files and modified files created for the implementation using JavaSoccer simulator

# I. PREFACE

## **1.1 Introduction**

Over the past decade agents has become one of the most important research topics on Artificial Intelligence (AI). This paradigm has been applied to several different areas of research and to solve many different problems.

Agents are becoming one of most active area in AI thanks to INTERNET applications. This technology is widely used in e-commerce and in some applications in the INTERNET [Greenwald and Kephart 99]. Although this is the best-known area, in general, in which agents are applied, there are many other and different fields: economics [Youssefmir, Huberman and Hogg, 1998], foraging [Balch 1998], football [Kitano et al 1997]....

Agent is a vague term, this is no already a clear definition for this term, and every researcher defines this term depending on their interests. Despite this lack of agreement on what an agent is, all works share the same idea. An agent is seen as a autonomous thing (program, system, car,...), intelligent and tends to communicate and co-operate with other agents.

## **1.2 Introduction to Research Work**

This research work tries to bring together several ideas from multiagent systems. This work will focus on intelligent physical multiagent systems, in what is called physical dynamical agents. This work brings together ideas from adaptative systems, ecosystems, resource allocation, dynamical systems... All these ideas will be applied to soccer domain

This works aims at analysing the impact of dynamics, as understood in general control theory [Dorf 1999], and diversity in a multiagent system, consisting of different robots playing soccer and adapting to the opponent depending on the opponent difficulty and their own physical body.

Some general terms will be defined here as understood in this work.

### **What is an agent?**

Although there is no clear definition for the term agent. Here there are two definitions, one general and a second one closer to this work and the one used in this work:

[Wooldridge 1999]: “ An agent is a computer program that is situated in some environment, and that is capable of autonomous acting in this environment in order to meet its design principles”. This definition can be extended to define which are the conditions for called an agent an intelligent agent [Wooldridge and Jennings 1995]:

- *reactivity*: intelligent agents are able to perceive their environment, and respond in a timely fashion to changes that occur in it in order to satisfy their design principles.
- *pro-activeness*: intelligent agents are able to exhibit goal-directed behaviour by taking the initiative in order to satisfy their design principles.
- *social ability*: intelligent agents are capable of interacting with other agents (and possibly humans) in order to satisfy its design principles.

One of the most important aspects in agents is social ability, social ability can be understood [Wooldridge 1999] as the necessity to negotiate and co-operate with other to achieve goals.

For this work a closer definition will be used, based on [Stone 1998], because of some similarities with this work (football domain). [Stone 1998] says, “an agent is an entity with perceptions, goals, cognition, actions, and domain knowledge, situated in an environment”. We would like to add to this definition that robot behaviour is conditioned by its physical body, namely, dynamical constraints. This means that some state variables (speed and acceleration) change through time according to physical laws.

### **What is a physical agent?**

A physical agent is an entity whose actions and co-operation with other agents are limited and conditioned by its physical features.

### **What is a physical dynamical agent?**

A physical dynamical agent is a physical agent, whose some of its features (speed, temperatures,...) can be described by means of differential equations.

## **1.3 Organisation of this work**

This work has been organised in several sections.

*Section II, State of the Art:* Several areas related to this work are surveyed. Especially those related to the field of agents.

*Section III, Guidelines of this work:* Built on the state of the art, the goals of this work are presented trying to work on something new.

*Section IV, Implementation:* Implementation is detailed in depth, explaining all the different aspects used in this work.

*Section V, Experiments:* Several experiments are performed in order to show how the implementation works and to find answers to the goals of this work.

*Section VI, Conclusions*

*Section V, Future work:* Future improvements and new research areas are presented.

## II. STATE OF THE ART

### 2.1 Physical Agents

Over many years AI has focused on the abstract manipulation of symbols [Simon 1969], this symbolic manipulation has been based on a physical reality, but this physical world has never been possible to represent it in a proper way for this reasoning. The system behaviour was the result of the symbolic reasoning. [Brooks 1990a] and [Brooks 1990a] proposed a new grounding for intelligent systems. The system relies on the information that is able to extract from the world, the world is sensed through the sensors and is connected to the world via actuators. The system behaviour is a emerging property from different behaviours, as a result of sensing the world and reasoning at different levels. In this new approach any symbolic representation has to be made taking into account sensors and actuators.

For [Brooks 1991] there are two central ideas that have led to this new approach in robotics:

- *Situatedness: The robots are situated in the world—they do not deal with abstract descriptions, but with the "here" and "now" of the environment that directly influences the behaviour of the system.*
- *Embodiment: The robots have bodies and experience the world directly –their actions are part of a dynamic with the world, and the actions have immediate feedback on the robots' own sensations.*

This new idea represented a turning point for AI and the starting point for a new way to understand intelligent systems and build new ones. Over the past decade this new approach has brought new advances in AI. Although initially and largely all these concepts have been applied to robotics, they have been also extended to other areas [Unsal 1997] and generally are called multiagent systems, multi robotic systems, autonomous systems... Agents are usually applied to autonomous systems without a central controller.

Within the scope of this work, one of the most interesting research has been done by [Zhang and Mackworth 1995]. In [Zhang and Mackworth 1995] they define a robot, although this can be extended to other systems, as a hybrid intelligent dynamical system, consisting of a controller coupled to its body. At the same time a robot is symmetrically coupled to a dynamic environment. To model the dynamics of they system and environment they developed Constraint Net (CN), a unitary framework to model dynamical systems. CN can deal with discrete and dynamical systems as well as synchronous and asynchronous event structures. CN is implemented as modules with I/O ports. Each module performs a transduction from its input traces to output traces, subject to the principle of causality: an output value at any time can depend only on the input values before, or at that time. A robot situated in an environment can be modelled as three machines, each of them modelled separately:

- Robot body
- Robot Controller
- Environment

Having a CN model and the required properties of a controller, specified as a set of constraints, it is feasible (if it is possible) to find in a automatic way a controller with the given specifications. [Zhang and Mackworth 1994].

In [Zhang and Mackworth 1998] these ideas are applied and implemented in a Robotic Controller for simulation.

According to [Asada, Kuniyoshi et al 1997], physical bodies play an important role in agents to achieve goals in dynamic real world, on which the traditional AI research has not paid so much attention. Physical bodies have an important role in enabling the system to interact with physical environments, that makes the system learn from the environment and develop its internal representation. The meanings of “having a physical body” can be summarised as follows:

- Sensing and acting capabilities are not separable, but tightly coupled.
- To accomplish the given tasks, the sensor and actuator spaces should be abstracted under resource bounded conditions (memory, processing power, controller, etc.).
- The abstraction depends on both the fundamental embodiments inside the agents and the experiences (interactions with their environments).
- The consequences of the abstraction are the agent-based subjective representation of the environment, and its evaluation can be done by the consequences of behaviours.

Even though we should advocate the importance of “having a physical body” it seems required to show how the system performs well coping with new issues in a concrete task domain.

Also in [de la Rosa, Garcia , Innocenti., et al 1999] the problem of having a physical body is considered. Based on [Shoham 1993], capabilities are represented about dynamics. Capacities based on dynamics mean having information about controllers and how some variables (i.e. speed) change through time. All this information is used in negotiation processes among several agents (deliberative agents) to perform a joint task. Some examples show the convoing of two vehicles or passing a ball [de la Rosa, Innocenti et al 2000]. [Oller, de la Rosa, del Acebo 1999] presents an agent architecture, called DPA, to design agents with dynamics. This architecture has three different levels: control, supervision and agent. Dynamic knowledge has a different representation for each level, being adapted it for a better use for each different level.

## **2.2 Self-Organisation and Emergent Behaviours**

Self-Organisation according to [Unsal 1993] and [Eoyang and Conway 1999] can be found in many fields: biology (insect societies, ecosystems), chemistry (thermodynamics), sociology (communication), economy, topology, physics... This idea of self-organisation are called by different terms and is studied in other fields: new science, chaos theory, complexity theory, complex adaptive systems research, non-linear dynamics, dynamical systems theory, and synergetic.

In [Yovits, Jacobi and Goldstein 1962] “self-organisation” is first defined as:

*“A self-organising system is a system that changes its basic structure as a function of its experience and environment”*

According to [Eoyang and Conway 1999], there are three conditions for the self-organisation of a system:

1. Differentiation. During this step a difference is recognised in the system and is amplified by some internal dynamic or some environmental influence.

2. Transforming feedback or coupling. During this step, a transforming bridge is built across the differentiation identified in Step 1.

3. Self-organisation. During this final step, the system moves to a new state in which the structural framework of the past is superseded by an emergent structure that responds to the feedback loops established in Step 2. However, there are no guarantees that self-organising will occur or that it will occur in ways anticipated or even for the good.

As a result of this self-organising process, emergent behaviours will appear. If these differences are quite important, some kind of competition among the agents will arise and the self-organised system will not show any dominant behaviour at the end of the process.

For [Rocha 1999]:

*“a self-organising is seen as the process by which systems of many components tend to reach a particular state, a set of cycling states, or a small volume of their state space (attractor basins), with no external interference. This attractor behaviour is often recognised at a different level of observation as the spontaneous formation of well organised structures, patterns, or behaviours, from random initial conditions (emergent behaviour).”*

One of the advantages of a self-organising system [Unsal1993] over a “pre-programmed system” is the use of simple programming required for the individuals, members of the environment. The collective behaviour of these individuals will have an adaptative character through interacting with the environment. This will allow to create complex behaviours, designing simple systems.

In agent systems [Mataric 94] emergent behaviour is characterised by:

*“ ..global states or time-extended patterns that are not explicited programmed in but result from local interactions between a system’s components,..... These types of systems are referred to as “self-organising” because of their apparent ability to create order”.*

Emergent Behaviours can be seen in some cases as a result of a self-organising process, as it has in which elements show a consistent behaviour, but also as a result of chaotic behaviours at the agent level [Hogg and Huberman 1991], that result in an emergent order. From a different point of view [Brooks 1989], but based on the same idea, thinks that it is possible that human behaviour is chaotic from which “order appears to emerge”.

One of the areas, in which the processes of self-organisation is common, is in biological systems. [Kauffman 1991] thinks that self-organisation is an “inherent property of some complex systems” and that some complex biological systems tend toward self-organisation.

Natural Systems have been analysed in depth and one of the properties that arise in this kind of environments is self-organisation, especially in insect societies. [Bonabeau, Theraulaz et al 1997a] claim “that complex collective behaviours may emerge from interactions among individuals that exhibit simple behaviours” in insect societies (ants, bees,..). [Reynolds 1987] is able to create a flock of birds, emerging from the use of three simple rules:

1. Collision avoidance
2. Velocity matching with nearby flockmates
3. Flock centering in attempt to stay close to nearby flockmates

Many works on self-organisation in biology has been adapted and used in AI. For example in [Bonabeu, Theraulaz at al 1997b] a task allocation model found in ants is applied to an agent system.

## **2.3 Ecosystems**

According to Encyclopaedia Britannica, an ecosystem is:

*The complex of living organisms, their physical environment, and all their interrelationships in a particular unit of space.*

*The principles underlying the study of ecosystems are based on the view that all the elements of a life-supporting environment of any size, whether natural or man-made, are parts of an integral network in which each element interacts directly or indirectly with all others and affects the function of the whole....*

Ecosystems are complex biological systems in which adaptation is an essential characteristic. [Devine, Paton and Amos 1997]. Some mathematical models of ecosystems simulate models of heterogeneous agents that evolve in a system, according to their fitness to some aspect of the ecosystem. Normally these agents compete for resources, usually food. The most successful species tend to create new ones, combining their own information and adding new one through Genetic Algorithm (GA) or other similar techniques [Mitchell 2000]. There are many examples in which this central idea can be found. Some examples are Herby in [Devine and Paton 1997] or Webmodel in [Caldarelli, Higgs and McKane 1998].

This idea of competition for resources has been used on what is called computational ecosystems. One of first works on that field was done by [Miller and Drexler 1988] in which they propose a model, based on ecosystems and markets mechanisms, for organising computation in large systems (allocation of storage capacity and processor time) and in which this mechanism will give rise to intelligent emergent behaviour. In [Waldspurger, Hogg et al 1992] the problem of assigning idle processor time in a network of heterogeneous workstations, in which an auction is stabilised to elucidate which process can use this spare processor time. Using a similar mechanism [Clearwater and Huberman 1993] design a system, in which a group of agents sell and buy cold air. [Ekblom and Astor 1996] survey these works in order to study a possible solution to efficient electricity distribution. In [Hogg and Huberman 1991] these agents compete for resources, but sense the environment in a imperfect manner. Although initially the behaviour of the resource allocation is chaotic the combination of several agents competing at the same market stabilise the system.

## **2.4 Heterogeneity in Multi-Agent Systems**

Not much work has been done in the field of multi-agent heterogeneous physical agents. Most work deals with homogeneous systems, usually involving robotics and some kind of autonomous vehicles. It is more frequent to find this kind of systems in software multiagent systems.

There are many advantages using several agents for solving a task instead of one single agent. [Ali 1999] summarises some reasons for using a multiagents robotic system:

- A larger range of possible tasks
- Greater efficiency
- Robustness
- Lower economic cost

- Ease of development

A central idea is the ability of a multirobotic system to perform any task that a single robot can do, but a single robot can not accomplish tasks that a multiagent system can.

[Parker 1998] claims that one of limitations in multirobotic agent system is the difficulty in designing a team of heterogeneous robots to carry out a given task. This decision normally relies on the robotic expert. One of the possible reasons to develop a team of heterogeneous robots is the possibility for decreasing robot complexity, splitting capacities among several robots. For [Grabowski, Navarro-Serment et al 2000] building a team of heterogeneous robots is cheaper and easier to maintain and debug. Although each individual robot can have limited capabilities they can accomplish the same tasks through co-operation and are more reliable. In case on robot fails few capabilities are lost and the robots can continue the task with the remaining robots, which normally have overlapping capacities. From a different point of view, in economics system, [Hong and Page 1998] claim that a group of heterogeneous agents, creating heterogeneity through different heuristics, can find solutions to very difficult problems.

[Lander 1994] classifies agent heterogeneity in Team Problem System:

Representation heterogeneity in

- Architecture
- Algorithm
- Language
- Inference engine
- Hardware requirements

Knowledge heterogeneity in

- Declarative knowledge
- Evaluation criteria for solutions
- Priorities
- Goals
- Capabilities

This heterogeneity influences system developments in many ways. In case of agent capabilities the system design requires some strategies for the co-ordination of interacting agents.

Some people has carried out some in depth work:

[Parker 1994] argues that is not possible to design all the capabilities in just one robot. For this reason is necessary to develop a group of robots with different capabilities to perform a task. She develops two agent architectures to cope with this diversity, these architectures are behaviour-based and are called ALLIANCE and L-ALLIANCE. ALLIANCE is an architecture for the co-ordination of several heterogeneous robots with overlapping capabilities, fault tolerant, adaptative action selection (based on a mental model) for small-medium sized robotic teams. L-ALLIANCE is an improvement from ALLIANCE.. In L-ALLIANCE she proposes tasks assignments according to the robot past experiences. From this information each robot estimates the time it needs to perform a given task. When a task has to be done, several robots compete for it, and the one, which is free, with a better fitness for this task, starts performing this task. Initially she proposes some random selection to explore agent capacities (time, energy).

Later [Parker 1998], based on its past works, outlines a methodology for automatic task allocation in a pool of heterogeneous systems. She thinks that some conditions are necessary for this design:

- a language to describe the specifications of any mission.
- Determine the metrics necessary to evaluate the performance of the mission (time, energy...).
- Describe robot capabilities and individual behaviours.
- Other: fault tolerance, efficiency...

Having a mission, the abilities necessary for this task should be found and the robots with the necessary abilities selected using some optimisation techniques to find the team with a less cost based on time, energy expenditure...

[Parker 1997] , using ALLIANCE architecture, presents CMOMMT the problem of cooperative multi-robot observation of multiple moving targets. She analyses the problem of placing a set of robots, with limited sensorial capacities, in a room to observe a set of multiple moving targets. Later in [Parker 1999], she outlines the adaptation of the robots to changes in the environment according to their capabilities among others.

[Lander 1994] presents a knowledge-based technology, called TEAM, in which a group of heterogeneous agent, each with an area of expertise, tries to solve a given problem with a set of specifications. Each of these agents works on a part of the problem, the overall solution is made up of solutions to subproblems. Each agent plays an organisational role in this search for a solution. An agent can choose to initiate a new design, extend a partial design or critique an existing plan. This process involves a framework where negotiations, ... can take place among these agents. All this procedure is applied to a steam condenser, as an example, in which a set of agents (heat exchanger, pump, motor, V-belt, Shaft, Platform) has to act in order to accomplish a set of specifications. Each agent has a set of capabilities and has to propose solutions to the specifications dealing at the same time the constraints. This work is extended later, [Nagendra, Lesser and Lander 1995] and [Nagendra, Lesser and Lander 1996], in a new framework called L-TEAM. L-TEAM allows the robots to learn their organisational roles from experience. After the learning process each agent knows what it has to do, which role has to play, according to the problem to be solved. This learning process show an improvement in result in comparison to results in TEAM framework.

[Balch 1998] in his dissertation analyses a system made up of identical robots, but with different behaviours. He tries to create this diversity of behaviours through reinforcement learning (RL) [Kaelbling, Littman and Moore 1996]. In order to analyse diversity, he introduces a new methods to evaluate diversity of behaviours, basing his work on [Shanon 1949]. This method computes a value for a group of robots, in which “0” means no difference and “1” maximum difference. In some experiments performed with several robots in a foraging task shows that greater diversity, with hand-coded and learnt diverse behaviours, is related to a lower performance.

Finally [Gerkey 1998] and [Gerkey and Mataric 2000] develop a multi-agent architecture called MURDOCH, in which tasks are allocated to robots according to the capabilities needed for performing the task. In this work capabilities for tasks and robots are known beforehand.

## **2.5 Consensus**

Consensus allows system to get a value, a piece of information,... combining several sources of information. Consensus is applied widely in AI, many times in combination with fuzzy logic [Zadeh 1965]. Some times it will be possible to merge all the sources and have one value (several sensors), in other situations one solution will be selected from a set of solutions proposed by different knowledge source (soccer), using a revised certainty.

The simplest techniques come from the field of statistics. These techniques include mean value, weighted mean and trim mean, in which the highest and lowest value are discarded [Chatterjee and Chatterjee 1987].

[Yager 1988] presents a more elaborate system for averaging sources called OWA (Ordered Weighted Averaging). In this case sources are ordered depending on their values, this values are weighted with a parameter associated to a position in the final ordering. OWA includes mean and trimmed mean.

[Torra 1998] extends OWA operator and develops WOWA (Weighted Ordered Weighted Averaging), in which the importance of the source is considered. In this technique each source are weighted by two factors, the position in the ordering and the importance of the source. WOWA is a generalisation of OWA and weighted mean.

[de la Rosa 1993] based his works on the idea of co-operative solving problems. Through co-operation among several knowledge base systems with the same goals to get better results it is possible than considering only one source of knowledge. For this purpose he developed a heuristic for co-operation of expert systems. The opinions of these expert systems are revised by two parameters: Prestige (P) and Necessity (N). P is a linear parameter that revises the certainties provided by the knowledge sources, according to the importance of KB. N is a non-linear parameter and it is a singularity of this problem. Non-linear transformations hold interesting properties for learning. Necessity can be understood as the necessity of information the system has, depending on the source. Results show that using N and P the results of co-operation are much better that using only P. P is an special case of N and P and also is a simple OWA and WOWA.

## **2.6 Football Domain**

Soccer has become one of the most important test-bed for multi-agent systems over the past decade. [Barman, Kingdon et al 1993] presented Dynamo, a test-bed to experiment in soccer with agents. In 1996 the first FIRA competition took place in Korea and in 1997 RoboCup in Japan. According to [Kitano, Veloso et al 1997] RoboCup is an attempt by using soccer game as a representative domain where wide-range of technologies can be integrated as well as new technologies can be developed. The RoboCup envisions a set of longer-range challenges over the next few decades.

Several works have been developed in this field. Within the scope of this work is specially interesting the one done by [Stone 1998]. This work deals with the problem of decision making through learning in soccer. He solves the problem of learning in such a complex system, learning at different levels, building the higher level on lower one. He applies Neural Networks to learn to intercept a ball, uses Decision Trees to decide to which player to pass, the player in possession of the ball ,and through Reinforcement Learning learns the team strategy.

Also the work done by [de la Rosa et al 1997] is interesting, in which they apply consensus techniques (see above) and agent based technologies using AGENT0 [Shoham 1993] to implement a soccer team.

To do research within the scope of soccer, to work under a controlled environment it is necessary. For this reason is extremely important to be able to work with simulators. One of the most interesting simulators based on Robocup is JavaSoccer [Balch 97]. JavaSoccer is implemented in Java, this means that it is portable between platforms, and its code its accessible and easy to understand, this means that the necessary modifications can be done to adapt the simulator to one's wishes.

[Johnson, de la Rosa and Kim 1998a] and [Johnson, de la Rosa and Kim 1998b] propose the realisation of several benchmarks to analyse the performance of several solutions implemented. These benchmarks have different purposes:

- To set rigorous scientific standard for research into robot football.
- To encourage teams to work on the same problems and to allow comparison without any explicit match of teams each other.
- To collect and publish data on robot control and ball control, and co-ordination.
- To enable scientific analysis of the performance of teams world-wide.
- To enable any particular team to gauge its performance against these standards.
- To provide a *simple* baseline from which new scientific benchmarks can be defined.
- To create proper families of benchmarks useful for adaptive/emerging techniques of AI.
- Testing of individual or collective behaviour.
- Testing with real or simulated robots in a real or simulated environment.
- Generated benchmarks have to be easy to build.

Base on these ideas a new benchmark has been developed, called Implicit Opponent [de la Rosa, Duhot and Muñoz 2000], which consists of a inclined field and a group of static opponents.

### III. GUIDELINES OF THIS WORK

Not much work has been done in the field of heterogeneous physical agents. Only [Parker 1994], [Parker 1998] and [Parker 1999] have done extensive work over the past years on this field. She has created heterogeneity, mainly, through sensor and actuator differences. This work aims at creating and studying the impact of heterogeneity in dynamics, as understood by [de la Rosa J. Ll, Garcia R, Innocenti B., et al 1999] and [Zhang and Mackworth 1995]. Dynamics plays an important role when a physical agent (robots, cars,..) is acting in its environment and co-operating with other agents. There is few work conducted in the field of assigning tasks to a pool of heterogeneous physical agents that have to perform a joint task in a co-operative way. [Parker 1994] tries to solve this problem with L-ALLIANCE framework. L-ALLIANCE assigns robots to tasks using time estimates to perform these tasks, this time estimates are computed during a period of training and are refined through experience. Later [Parker 1998] outlines a method to allocate tasks to robots according to sensors, actuators, behaviours, communication and co-operation, but she does not consider dynamics. It seems that this method has not been tested yet and not further work has been developed. [Gerkey and Mataric 2000] uses a publish/subscribe messaging method to allocate agents to tasks, this allocation is done considering agent capabilities. In case two or more agents are able to perform the same task, a metric (time, energy,...) is applied to elucidate which agent is going to perform the task. No one has considered yet, in this problem of task allocation, possible changes in physical capabilities. [Parker 1999] acknowledges this problem and thinks that some kind of adaptation is needed to address this problem.

Considering the state of the art, this work aims at:

- **Creating heterogeneity through dynamics and studying its impact in co-operation**

Dynamics heterogeneity will be created based on the following idea: “ Agents designed to be fast will have low accurateness and agents designed to be slow will have high accurateness. ”. This design principle will affect other physical aspects of the agent: stability, persistence..., that will affect co-operation among agents

- **Allocation of roles to agents through a self-organisation process.**

This self-organising process will be built on the dynamics of ecosystems described by [Hogg and Huberman 1991]. This formulation allows to frame the preference of a group of agents over a set of choices, depending on the rewards they get from each possible option.

- **Studying of convergence to roles in the self-organisation process and its relationship with its body.**

This emergent process can lead to an emergent situation, in which a set of agents, which initially did not show any preference for one of the roles, tend to use one of the resources..

- **Studying changes in the dynamic behaviour of the agent.**

Changes in dynamic can be understood as a change of a robot, because it failed, or changing its behaviour through the time (running faster, slower). It is interesting to study how the system evolves when one of the agents changes its dynamics

- **Studying the impact of the heterogeneity in the process of self-organisation**

Self-organising mechanism is designed for a group of heterogeneous agents, but also homogeneous agent system can be tested.

- **Multi-agent systems problem design.**

This methodology can allow a multi-agent system designer to hand-code knowledge about a problem, for a group of agents that have to perform some joint task. In this problem the designer does not know how assign tasks to agents, in a optimal way, or he/she wants to allocate a subset of tasks to a sub-set of agents and for the remaining tasks and agents, he/she does not know how to match tasks and agents.

## IV IMPLEMENTATION

### 4.1 INTRODUCTION

Several aspects of this proposal will be implemented and tested in football domain. Some past works [Stone 1998] have shown that football domain offers an interesting environment for multi-agent co-operation, in which interesting results has been achieved. It is also possible to apply results to other multi-agent domains (network management, e-commerce,...).

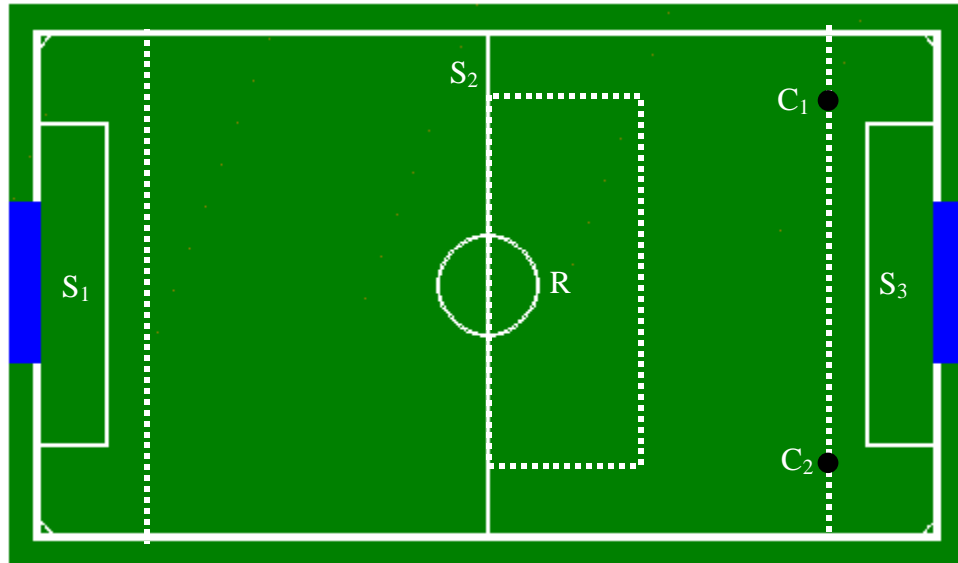
[Johnson, de la Rosa and Kim 1998] lay out some conditions to conduct research in football domain. It is important to define standard benchmarks, in order to be able to compare results from several runs and the impact of different strategies. Also it is important to be able to work under controlled conditions. If these conditions are not well known (noise,...) , results might not be proper for analysis. For this reason all the experiments will be done in simulation, using JavaSoccer [Balch 1997] in which conditions are well known and can be controlled, and in a new benchmark, called implicit opponent, that will be defined later [de la Rosa, Duhot and Muñoz 2000].

### 4.2 JavaSoccer

JavaSoccer [Balch 1997] is a soccer simulator that conforms to RoboCup rules. It has been written in Java and can be run in almost every computer. One of the most important advantages is the possibility of modifying the simulator to adapt it to one's needs. It is also easy to implement strategies and different agent behaviours. JavaSoccer will be used in this work for these reasons and its simplicity.

Some modifications have been introduced in order to make it useful to this work.

- In the original program ball is always placed at midfield, after 60 seconds time have elapsed without the ball being kick or after 5 seconds have elapsed after the last time the ball was moved by a player. In order to conform to RoboCup rules and benchmark definition (see below in benchmark) some modifications have been introduced (Fig.1):
  1. Each time the ball gets stuck or it is stopped below a given  $x_1$  position ( $S_1$ , own goal) the ball is repositioned randomly at midfield ( $R$ ).
  2. After 60 have elapsed the ball is repositioned randomly at midfield ( $R$ ).
  3. Each time the ball gets stuck or it is stopped above a given  $x_2$  position ( $S_3$ , opponent goal) the ball is repositioned at the corners ( $C_1, C_2$ )
  4. In any other event the ball is repositioned 20 cm off (y-axis) the last position ( $S_2$ ).



**Figure 1. Field zones for ball repositioning**

These modifications have been introduced in class *GolfBallS.java* that substitutes class *GolfBallSim.java*.

- A constant acceleration has been included in the model of the ball in order to simulate an inclined field, according to the benchmark (see below benchmark). This acceleration changes depending on the inclination of the field. The equation for this acceleration is:

$$a = g \cdot \sin(\alpha) \quad (1)$$

$g$  is the gravitatorial acceleration  
 $\alpha$  is the inclination of the field

These modifications have been also included in class *GolfBallS.java*.

- According to Robocup regulations, only one player per team can be at the same at each goal area, only accidental passing is allowed. In any other event a penalty kick is called. In order to conform to Robocup regulations JavaSoccer (modified version) only allows one player at the same time at the goal area. This rule has been hand coded and stops any additional robot (second robot) that tries to enter the area. These players stay still next to the goal area, until they decide to move to a legal position.

These modifications have been included in class *SimulationCanvass.java*, *SimulatedObjectNew.java* and *SocSmallSimNew.java* that substitute *SimulatedObject.java* and *SocSmallSim.java*.

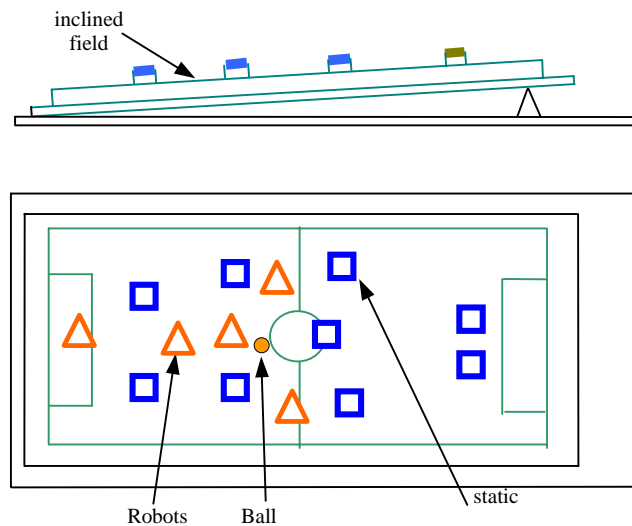
- Description file *robocup.dsc* has been modified in order to be able to specify robot features (size, speed, kick speed) in an accessible place an easy to change. These changes in *robocup.dsc* has led to changes and additional modifications in other classes (*SimulationCanvas.java*, *Simple.java*, *SimpleInterface.java*, *SocSmall.java*, *SimulatedObjectNew.java*) in order to have access to this information when creating a new object for a new player.
- A special strategy has been coded for the static players, necessary for the benchmark (see below), called *Ghost.java*. *Ghost.java* has no decisions programmed, as a result of it players do not move from their initial positions.

- All players report their rewards values each  $T_w$  seconds to class *SimulationCanvas.java*, in order to update population values (see implementation). As a result of it new public variables have been added to *ControlSystemS.java*.
- Players are repositioned at the initial positions every time a goal has been scored. Initially these players are randomly repositioned, but after a given time these players are repositioned depending on a parameter (see example below). A method *SetKOff* has been created in *SocSmallSimNew.java*.

### 4.3 Benchmark Generator: Implicit Opponent

#### 4.3.1 Introduction

The principle of this benchmark is to use a team of robots against a fictitious (implicit) opponent that is modelled by an inclined field with variously shaped static obstacles (Fig.2 and Fig.3). For this reason this benchmark is called implicit opponent. The goal of the team will be to play against this implicit opponent and for instance play a normal match, or do other type of plays. This will be further discussed later.



**Figure 2. Proposed benchmark set-up**

This benchmark is a generalisation of benchmarks that were proposed by [Johnson, de la Rosa and Kim 1998b]. The expected interesting features of this benchmark are:

- The implicit opponent's team behaviour is clear and predictable.
- Any number of robots from 1 can evolve in any F-xxx platform both in real and in simulation. The F-xxx contains any RoboCup and FIRA platforms.
- It contains requirements for control, co-ordination, and co-operation.
- Continuous interest in the evolution of the benchmark: this is good for continuous and dynamic movements. Its provides with good data for learning and emergent behaviours.
- These good features allow a good reference benchmark so that all existing teams can be evaluated/compared in a robust and general way.

### 4.3.2 Formulation of the Benchmarks Generator

This benchmarks generator will be based on the principle of an implicit opponent.

### 4.3.3 Purpose

The purposes of the benchmarks generator are:

- To set rigorous scientific standard for research into robot football.
- To encourage teams to work on the same problems and to allow comparison without any explicit match of teams each other.
- To collect and publish data on robot control and ball control, and co-ordination.
- To enable scientific analysis of the performance of teams world-wide.
- To enable any particular team to gauge its performance against these standards.
- To provide a *simple* baseline from which new scientific benchmarks can be defined.
- To create proper families of useful benchmarks useful for adaptive/emerging techniques of AI.
- Testing of individual or collective behaviour.
- Testing with real or simulated robots in a real or simulated environment.
- Generated benchmarks have to be easy to build.

### 4.3.4 Description of Parameters

Every benchmark **B** with a name "name" will be defined by the tuple  $\langle \mathbf{F}, \mathbf{P}, \mathbf{P}_r, \mathbf{P}_o, \mathbf{T}_e, \mathbf{F}_b, \mathbf{C}, \mathbf{J} \rangle$ . The performance of the team at every experiment on the benchmark will be contained in instances of **J**. All these terms are defined as follows:

- **The Formula**

The formula **F** is both the 2-D terrain and type of football robots both in real and in simulation that will be used to implement a benchmark. We will use the nomenclature of RoboCup. So **F** are:

- **F-1 Simulator League**
- **F-180 Small Size League**. Small real robots.
- **F-2000 Middle Size League**. Middle real robots.
- We define the simulation league that uses JavaSoccer as the **F-2 Simulator League**.
- We define the MIROSOT league as the **F-100**

The origin of every formula is (0,0) metres, point that will be situated in the top left corner. In the future surely there will be other formulae. The formula **F** defines the limits of the terrain, and the type of football robots. For instance, today the size of the terrain in F-180 and F2 is a Ping-Pong table, F-2000 is of 16 Ping-Pong tables, F-1 is the size of 105x68 metres, F-100 is a little more than half Ping-Pong table. However, our *implicit opponent* benchmark will be conceived to be equal and perfectly used in any formula. This is very important, especially to prototype results in simulation (F-1) and then easily exports them to the other real implementations (F-180 and F-2000).

- **The Angle of Inclination**

It consists in inclining the terrain of the platform formula **F** in some angle **P**. In this benchmark generator the angle will be normalised in a set of positions using radians in a

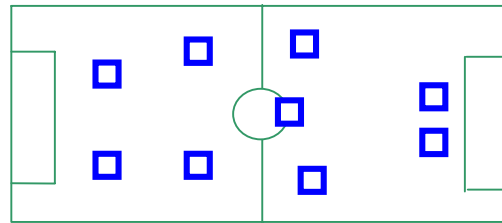
range that clear dynamics of the ball and robots will be observable. For instance  $\mathbf{P} \in [0, 0.2]$  radians. This makes the ball move continuously down the angle against the goalkeeper of the tested team. The behaviour of a free ball is simply its dynamics driven by the gravity's force. This will introduce continuous movement of the ball, which will behave as if a hidden *implicit opponent* team of agents was moving there.

- **The Pattern of Evaluated Robots**

The pattern of evaluated robots  $\mathbf{P}_r$  consist of a certain number of robots  $N_r$  of the evaluated team, that will be any number from 1 with  $N_r \in [1, \infty]$ , the shape  $\mathbf{S}_r = \{\text{round, square, others}\}$  the robots are made of, the total area occupied by the robots  $\mathbf{A}_r$  and finally a pattern of their initial conditions, that is the initial geometric positions on the 2-D terrain, that will consist in a list of  $\langle \mathbf{x}, \mathbf{y}, \theta \rangle$  positions, with units in metres and refereed to the same origin of the formula  $\mathbf{F}$ . Thus, a pattern of evaluated robots  $\mathbf{P}_r$  will be defined by the tuple  $\langle N_r, \mathbf{S}_r, \mathbf{A}_r, \langle \mathbf{x}_1, \mathbf{y}_1, \theta_1 \rangle, \dots, \langle \mathbf{x}_{N_r}, \mathbf{y}_{N_r}, \theta_{N_r} \rangle \rangle$

- **The Pattern of Opponents**

The Pattern of Opponents  $\mathbf{P}_o$  consists of a certain number of obstacles  $N_o$ , what shape  $\mathbf{S}_o = \{\text{round, square, others}\}$  the obstacles are made of, total area occupied by the opponents  $\mathbf{A}_o$  and finally a pattern of their geometric positions on the 2-D terrain, that will consist in a list of  $\langle \mathbf{x}, \mathbf{y} \rangle$  positions, with units in metres and refereed to the same origin of the formula  $\mathbf{F}$ . Thus, a pattern of opponents  $\mathbf{P}_o$  will be defined by the tuple  $\langle N_o, \mathbf{S}_o, \mathbf{A}_o, \langle \mathbf{x}_1, \mathbf{y}_1, \theta_1 \rangle, \dots, \langle \mathbf{x}_{N_o}, \mathbf{y}_{N_o}, \theta_{N_o} \rangle \rangle$



**Figure 3. An Example of a pattern of opponents, with 9 obstacles**

Note that there will be  $N_o \in [0, \infty]$  obstacles.

In the case of  $N_o=0$ , if the interpretation of  $N_o$  is the number of opponents, then  $N_o=0$  is only possible when  $\mathbf{P}=0$  (the case that there is no opponent).

#### 4.3.4 The Time to Do the Experiment

It is necessary to limit the time  $\mathbf{T}_e$  to do experiments in the benchmark for their better comparison. Units are in seconds. Its value can range  $[0, \infty]$ .

#### 4.3.5 The Features of the Ball

The features  $\mathbf{F}_b$  related to the ball have to be declared: type of the ball  $\mathbf{T}_b$  {light = ping-pong, average = golf, heavy = pin-ball}, whether the ball can or cannot rebound in walls,  $\mathbf{CAN} = \{\text{rebound, not rebound}\}$ , the random function  $\mathbf{R}$  that will randomly generate  $\langle \mathbf{x}, \mathbf{y}, \theta \rangle$  positions (units in metres and origin the same of formula  $\mathbf{F}$ ) to place the ball in the field whenever the ball get stuck and the seed  $\mathbf{SD}$  for the random function, which will be an integer. Then  $\mathbf{F}_b$  will be defined by the tuple  $\langle \mathbf{T}_b, \mathbf{CAN}, \mathbf{R}, \mathbf{SD} \rangle$

#### 4.3.6 The Constraints

A first simple constraint for coercing the co-operation skills of the evaluated team is introduced. It consists of limiting the time  $T_p$  any agent is allowed to have the ball under its control before getting rid of it or passing it to another agent. This time will be  $T_p \in [0, \infty]$  seconds.

A second constraint is the norm of velocity of robots of the evaluated team,  $\|V_r\|$ , that is the norm of the angular and linear speed of robots. Declaring this constraint is important since there will be some types of benchmarks where this magnitude will be lower than the maximal velocity of the ball  $\|V_o\|$  "driven" by the implicit opponents. In this situation,  $\|V_r\| < \|V_o\|$ , co-operation is strictly necessary. In the situation  $\|V_r\| < \|V_o\| / N_r$ , that is the maximal velocity of any robot of the evaluated team is lower than the maximal velocity divided by the number of evaluated robots, then the co-operation of all robots will be strictly necessary. This measure will be in units of metres/second and ranges  $[0, \infty]$ .

A third constraint is the norm of velocity of the shooting velocity  $\|V_s\|$  for any robot of the evaluated team. This measure will be in units of metres/second and ranges  $[0, \infty]$ .

A third constraint is a prohibited zone  $Z$  that consists of a square where the robots cannot enter in. Units are in meters, and the square is declared by its four corner points  $Z (<x_1, y_1>, <x_2, y_2>, <x_3, y_3>, <x_4, y_4>)$ . The origin is the same origin of the formula F we were using. Then the constraints  $C$  is a tuple of  $< T_b, \|V_r\|, \|V_s\|, Z >$

#### 4.3.7 The Performance Index

The performance index  $J$  is based on measurable features, which could include the following:

1. The total amount of goals:  $G_i$  for the implicit opponent,  $G_e$  for the evaluated team.
2. The differential amount of Goals  $G_e - G_i$
3. The (average  $aTis$ , best  $bTis$ , worst  $wTis$ , etc) **time** that the implicit opponent team needs to score (that is, the time the evaluated team receives a score):
4. The (average  $aTes$ , best  $bTes$ , worst  $wTes$ , etc) **time** the evaluated team needs to score.
5. The average distance  $aDb$  and standard deviation that the evaluated team maintain the ball from its local goal area.
6. The ratio  $R_p$  of the number of successful passes versus the unsuccessful passes.
7. The (average  $aTet$ , best  $bTet$ , worst  $wTet$ , etc) **time** that the evaluated team can maintain the ball in a target area.
8. Others to be declared in the future.

The performance index  $J$  will be generated by the joint evaluation of these measurable features.

#### 4.3.8 Summary

$$B \text{ "name" } < F, P, P_r, P_o, T_e, F_b, C, J > \quad (2)$$

- $P$  is an angle  $\in [0, 0.2]$  radians.

- $P_r$  is a pattern of evaluated robots and is the tuple

$$< N_r, S_r, A_r, <x_1, y_1, \theta_1>, \dots, <x_{N_r}, y_{N_r}, \theta_{N_r} >> \quad (3)$$

- $\mathbf{P}_o$  is a pattern of opponents and is the tuple  

$$\langle \mathbf{N}_o, \mathbf{S}_o, \mathbf{A}_r, \langle \mathbf{x}_1, \mathbf{y}_1, \boldsymbol{\theta}_1 \rangle, \dots, \langle \mathbf{x}_{N_o}, \mathbf{y}_{N_o}, \boldsymbol{\theta}_{N_o} \rangle \rangle$$
 (4)

- $\mathbf{T}_e$  is a time to do experiments  $\in [0, \infty]$ .

- $\mathbf{F}_b$  contains the features of the ball and is the tuple  

$$\langle \mathbf{T}_b, \mathbf{CAN}, \mathbf{R}, \mathbf{SD} \rangle$$
 (5)

- $\mathbf{C}$  is the constraints and is the tuple  

$$\langle \mathbf{T}_b, \|\mathbf{V}_r\|, \|\mathbf{V}_s\|, \mathbf{Z} \rangle$$
 (6)

- $\mathbf{J}$  is the performance index

## 4.4 Ecosystem Formulation

### 4.4.1 Introduction

[Hogg and Hubermann 1991] proposed a mechanism to model interaction and competition for resources among a group of agents. They showed how this model worked with simple but interesting examples. They started modelling resource choice for a group of agents

The resource choice is modelled by the following equation:

$$\frac{df_r}{dt} = \alpha(\rho_r - f_r) \quad (7)$$

$f_r$  is the fraction of agents using resource  $r$ , in this example there are two resources.  $\alpha$  is the rate at which agents re-evaluate their resource choice and  $\rho$  is the probability that an agent will prefer resource 1 over 2.  $f_r$  is also called a population of agents, this term is widely used in this work.

In order to obtain the value of  $\rho$ , the difference of performance obtained by using resource 1 ( $G_1(F(f_1))$ ) and resource 2 ( $G_2(F(f_2))$ ) is computed. These performances may be obtained evaluating real actions, as done in this work, time required to complete the task, accuracy of the solution... In this paper authors use an algebraic function (depending on agent populations) to model performance based on the following idea:

The performance of a group of agents using the same resource can increase, when introducing new agents, up to a certain value, due to co-operation among agents. From this amount of agents on performance diminishes because of the increasing number of conflicts (among this great number of agents) and they tend to waste efforts. In this example these two concepts are modelled with this two equations:

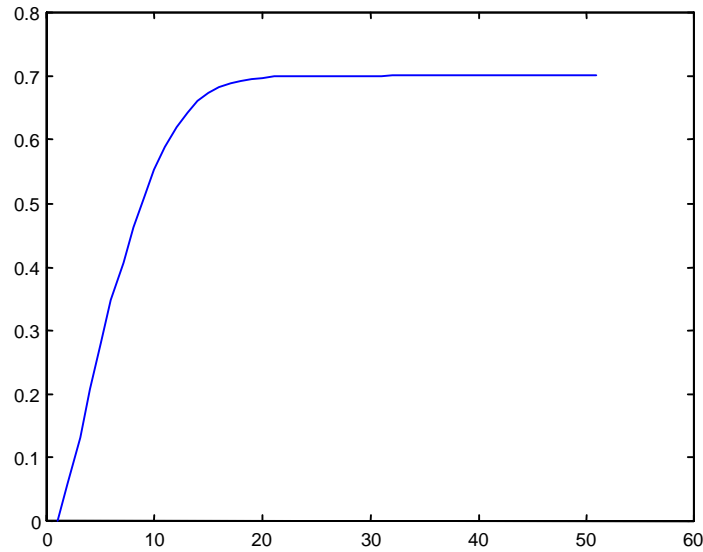
$$\begin{aligned} G_1 &= 4 + 7 \cdot f_1 - 5.333 \cdot f_1^2 \\ G_2 &= 7 - 3 \cdot f_2 \\ f_1 &= 1 - f_2 \end{aligned} \quad (8)$$

The more different performance, the more preference on the resource 1. And inversely resource 2 is with respect to resource 1. The uncertainty ( $\sigma$ ) is modelled as a typical

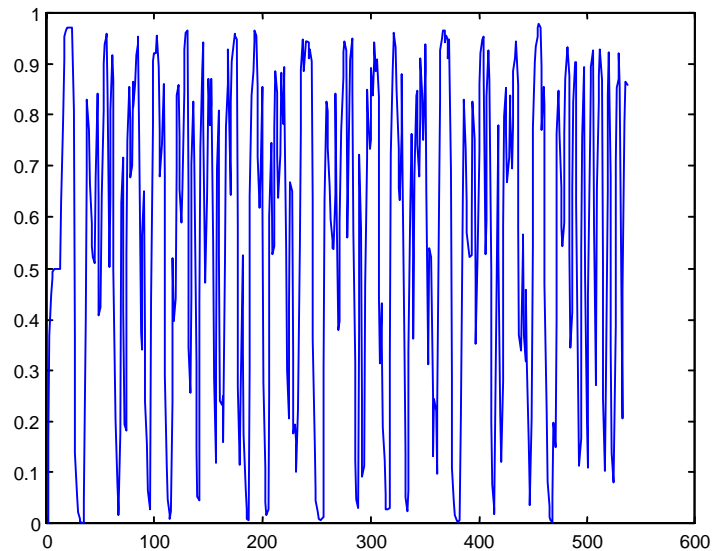
deviation on performance to decide when clearly resource 1 drove the agent to a better decision and performance than resource 2.

$$\rho = \frac{1}{2} \cdot \left( 1 + \operatorname{erf} \left( \frac{G_1(F(f_1)) - G_2(F(f_2))}{2\sigma} \right) \right) \quad (9)$$

Using this simple equation they do several experiments. When working with perfect information, information is up to date and available at any moment, the resource choice after a transient period stabilises (Fig 4). When this information is delayed, the agents have access to the state of the system  $k$  sample times ago, the system shows a chaotic behaviour, in which the system is continuously oscillating showing a chaotic behaviour (Fig 5).



**Figure 4. One agent with perfect information**



**Figure 5. One agent with delayed information**

This resource choice problem is extended to a group of agents. Each of them has a different perception of the state of the system, this means that each agent perceives this information

with a different delay of time. At the same time a new term is introduced, in order to reward those agents that perform better than others and penalise those that perform worse.

These are the resulting equations:

$$\frac{df_{rs}}{dt} = \alpha(f_s^{type} \rho_{rs} - f_{rs}) + \gamma(f_r^{res} \eta_s - f_{rs}) \quad (10)$$

$$\frac{df_s^{type}}{dt} = \gamma(\eta_s - f_s^{type}) \quad (11)$$

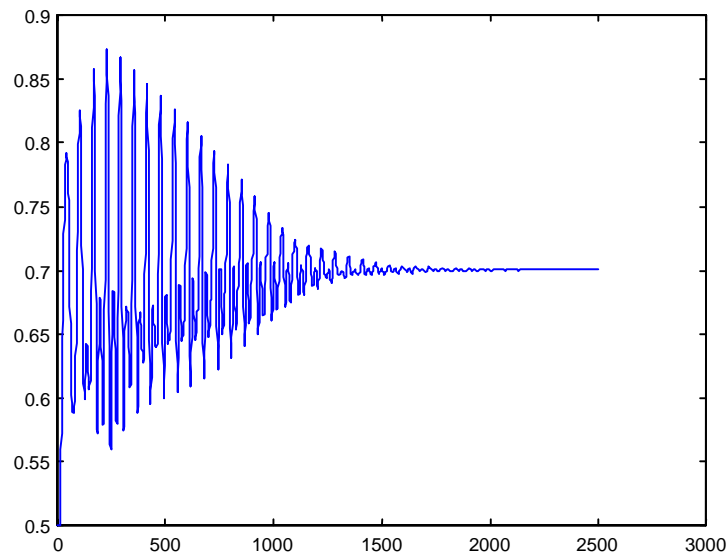
$s$  denotes type of agent. In this example each type of agent, as it has been said, has a different perception of the system, type of agent  $k$  perceives the system with a delay of  $k$  samples time. The fraction of agents of type  $s$  ( $f_s^{type}$ ) is changed through the second equation, according to their performance. This relative performance, how good/bad with respect to the average, is computed in  $\eta_s$ :

$$\eta_s = \frac{f_{1s} \cdot G_1 + f_{2s} \cdot G_2}{f_1^{res} \cdot G_1 + f_2^{res} \cdot G_2} \quad (12)$$

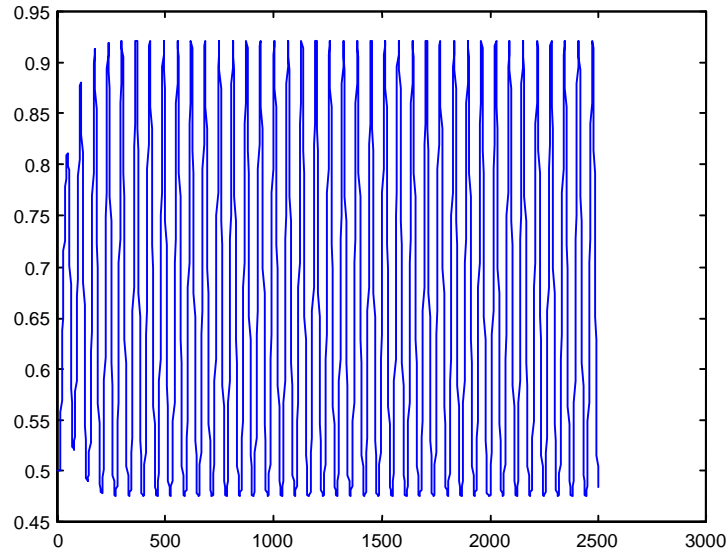
$f_r^{res}$  is the fraction of agents, considering all types of existing agents, using resource  $r$ .

The value of  $\rho$  is computed for each type of agent using delayed information ( $k, k+1, k+2, \dots$  samples of time) about the performance of the system.

$\gamma(f_r^{res} \eta_s - f_{rs})$  reward those agents that are doing better, in which  $\gamma$  is a rate similar to  $\alpha$ . This reward function is responsible for the stabilisation of the system together with the diversity of agents (Fig. 6). In any other case the system (resource choice) presents a chaotic behaviour, as in the example with one agent (Fig. 7). With this new term the system is able to reach an stable configuration.



**Figure 6. Stabilisation of resource choice in a group of agents**



**Figure 7. Oscillating resource choice**

A previous work used this ideas to stabilise a physical system (inverted pendulum), using two resource information (software agents). These two knowledge sources were responsible for controlling an inverted pendulum, using as inputs (position and velocity of the cart and pole). The information provided for these two resources were combined using a consensus technique (see consensus). One of the parameters of this consensus technique was related to the value of  $f_r$ , which changed depending on real performance. Although no results were obtained, this work has been very useful for this work, which is built on these both mentioned works.

This work builds on these previous equations, although some modifications have been introduced to adapt it to a different problem.

- The choice equation will be adapted to cope with five resources (also called roles).
- Performance will be obtained from real interaction with the environment.
- Diversity will be created using different physical features to design each agent.
- The state of the system (performance) will be evaluated after a window of  $T_w$  seconds and population values will be updated after this time.
- The changes in populations are coupled to the perception that every agent has of the co-operative world.
- Stability will be analysed from the point of view of resource usage and convergence to roles, but not how populations evolve.

#### 4.4.2 Choice equation

These equations are used to increase or diminish the preference of one agent for each of the resources (roles). This preference affects the perception of the co-operative world of each agent (see below), coupling the population values ( $f$ ) to consensus parameters.

In this case resources ( $r$ ) are roles and the diversity of agents ( $s$ ) are created using different physical features for each agent.

These are the resulting equations, adapted from the original work. These equation can deal with  $N \in [1, \infty]$  agents.

$$\frac{df_{rs}}{dt} = \alpha (f_s^{type} \rho_{rs} - f_{rs}) \quad (13)$$

$$\frac{df_s^{type}}{dt} = \gamma (\eta_s - f_s^{type}) \quad (14)$$

$$\rho_{rs} = \frac{G_{rs}}{\sum_{r=1}^R G_{rs}} \quad (15)$$

$$\eta_s = \frac{\sum_{r=1}^R G_{rs}}{\sum_{s=1}^S \sum_{r=1}^R G_{rs}} \quad (16)$$

All equations have been modified for this new problem. The main changes are due to the impossibility of using the equations, as they were formulated. For this new approach  $\rho$  as it is implemented in [Hogg and Huberman 1991] only can cope with two resources, now there are five resources. Although the new equation is not the same as the original, it loses the non-linear features, it gives a higher preference to those resources that perform better (non-linear features are recovered later at a different level). The main equation (Eq.13) does not have the reward term and the way  $\eta_s$  is computed has changed, although it can cope with five resources. There is a main reason for these changes. In this problem performance is computed considering the interaction of agents with the environment, but this performance has nothing to do with resource usage, as in original work. This new approach leads to a situation, in which parts of these equations do not make sense. There is not any close relation between the value of  $f_{rs}$  and the real performance. For this reason, and as a first approach to deal with this new problem,  $\eta_s$  is computed taking into account only performance values for each agent and the reward term is removed.

#### 4.4.3 Agent Performance

As it has been said agents performance is evaluated considering the actions taken by the agents and their impact in the environment, where these agents act. Performance is computed for each agent  $s$  and resource  $r$  in a temporal window of  $T_w$  units of time. These values ( $G_{rs}$ ) are computed taking into account several aspects:

Some actions are rewarded taking into account the current role for the player, other actions are rewarded depending on the position on the field. There are several reasons for this discrimination:

- Some abilities are barely correlated with a role, a player acting as defender with a good dribbling ability perhaps should play as a forward. Although defender must be able to control the ball, perhaps it might better contribute to the team in a different role.

- Some players may tend prematurely towards one role, in this situation players only will get rewards mostly from one role, when this player is much better acting in a different role. This assignment of rewards prevents the agent from quick convergence.

No rewards are given when the ball gets stuck, because this is a special situation and is abnormal to soccer.

Several actions and situations are evaluated to compute performance:

### 1) Set point

Roles propose actions, and actions give set points to agents. If the agent is at the given point, at the time a new decision is taken a reward is given. There is one problem in this method: Some resources proposed easier set points than others. For example goalkeeper stays at the same position for long periods of time. In order to have a fairer system to compute these rewards two aspects have been introduced:

- Depending on the decision taken a part of this reward is given, when the agent is within a given distance. This distance depends on actions.
- For defensive roles in defensive situations, part of these rewards (for *Defend* and *Cover\_goal* actions) are given depending on the part of the goal covered by the player, considering the angle covered by the defender (angel with the ball) and the angle of the ball with the goal-posts.
- Some other types of rewards offset some of these differences.

This reward is computed each time a decision is taken, this means lots of times in each window time. For this reason the maximum reward for each decision is very poor.

### 2) Position of the player and role

Any time an agent takes a decision its position and the role are compared. For each role a zone are depicted (Figure 8).

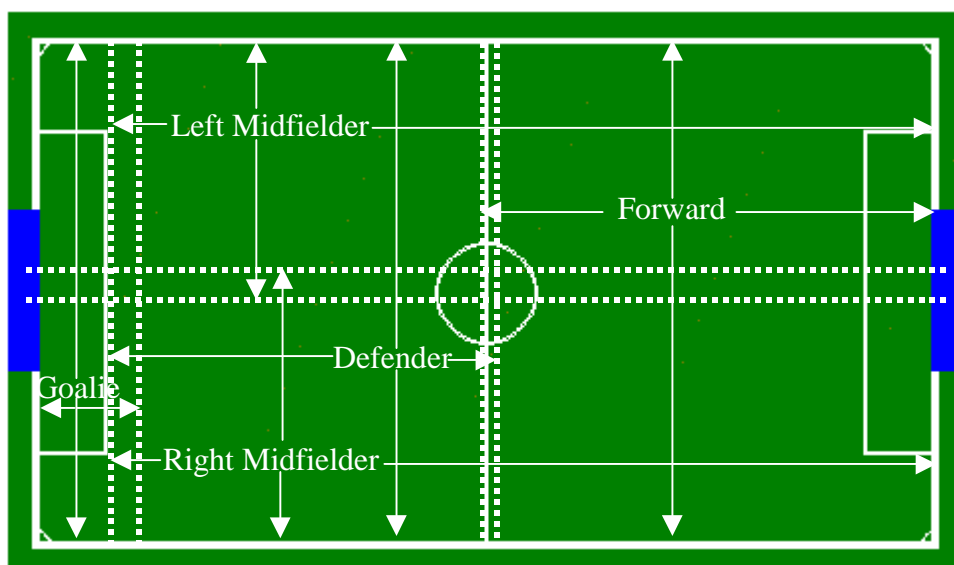


Figure 8. Role zones

If the position of the robot is in the zone, belonging to the role that it is using, a reward is given. Here parts of these rewards are also given when the agent is within a fixed distance from this area. This parameter helps to reward those agents that are acting coherently with one of the roles. One problem arises when these rewards are too high, this can cause too fast convergence of agents towards roles, and a wrong one. For this reason it is necessary some testing previously. Also here rewards should be poor because this reward is computed every time a decision is taken.

### 3) Dribbling ability

Only midfielders and forwards get rewards from this ability. It is very important for forwards to be able to control the ball, for midfielders it is also important but not so crucial as for forwards. For this reason rewards for forwards are higher than for midfielders. This reward is given depending on the position of the player in the field, in this case the role the agent is using is not considered (Figure 9). Dribbling is considered when a player is able to bring forward the ball in a controller manner, each time that happens a reward is given to the agent and given to the role depending on the position

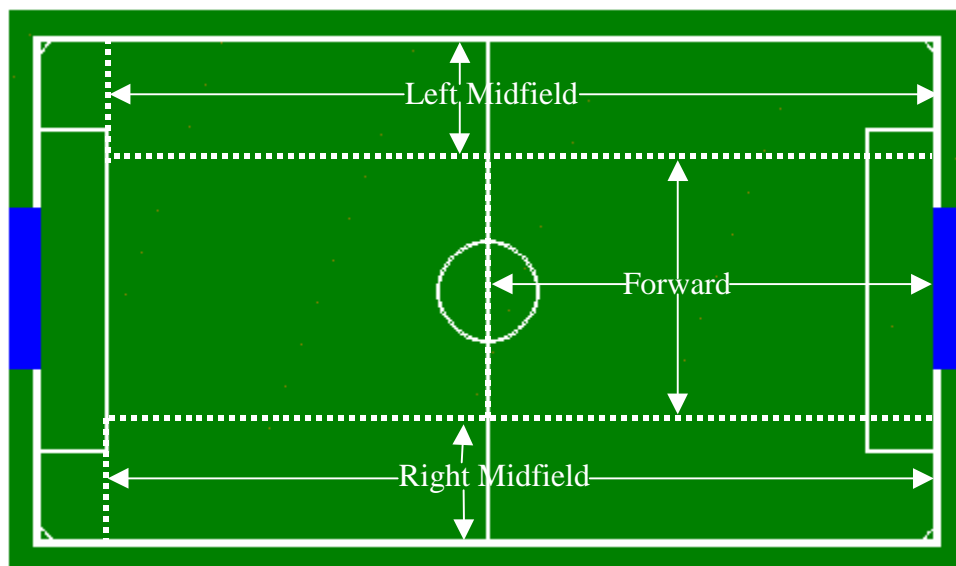


Figure 9. Dribbling ability zones

### 4) Goalkeeper ability

Goalkeeper ability is evaluated when it is inside the goal area or it is the last player and it is within a fixed distance from the goal area. Two actions are given reward:

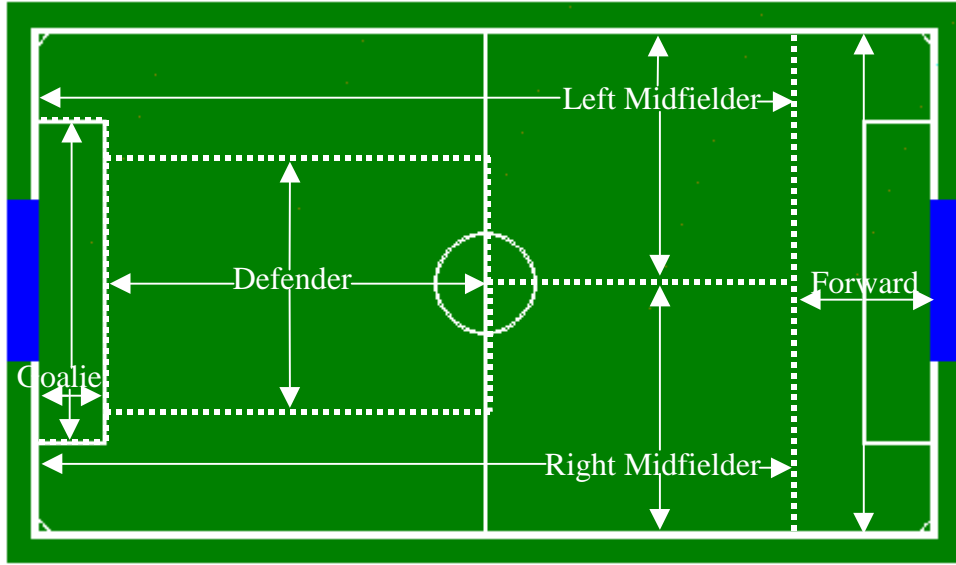
- The ball is falling down (in this benchmark) and it is able to bring the ball forward.
- The ball is falling down towards the goal and saves this shot (changing the direction of the ball to a no dangerous direction).

This ability is rewarded also considering the position of the player and not the role.

### 5) Ability to regain the ball

This ability is the same as the first action in goalkeeper ability. Rewards are given according to the position of the player in the field. This ability to regain the ball is much more

important in defensive roles than offensive roles. For this reason a higher reward is given to the first ones (Figure 10).



**Figure 10. Regain ball zones**

#### 6) Goals scored/received

Goals scored/received are used to increase/diminish the current rewards. If a goal is scored, at the end of the window time, the one that scored the goal increases its reward according to this equation:

$$\begin{aligned}
 Forward &= G_{forward} \cdot ngoals^{n1} \\
 Midfielder &= G_{midfielder} \cdot ngoals^{n2} \\
 n1 &> n2
 \end{aligned}
 \tag{17}$$

Goals received are used to diminish rewards at the end of the window time. Goalkeeper at the goal and close defenders are penalised according to this equation, if they were present at the time the goal was scored

$$\begin{aligned}
 Goalkeeper &= G_{goalkeeper} \cdot ngoals^{-n1} \\
 Defender &= G_{defender} \cdot ngoals^{-n2} \\
 n1 &> n2
 \end{aligned}
 \tag{18}$$

At the end of the window time goals scored and received are computed for each agent, depending on their positions (goals received) and goals scored.

#### 4.4.4 Ecosystem and co-operative perception

Equations (13-16), as it has been analysed, tend to increase the value of  $frs$ , for agents of type  $s$ , to those roles/resources that have better rewards associated. In order to deal with the

idea of preference for one role, the value of  $f_{rs}$  is used to assign the parameter Prestige ( $P$ ), for the certainty revision process. The certainty provided by the rules is revised by a value of Prestige depending on the role and player (see consensus technique). Higher values of prestige mean more chances to use the role.

$$P_{rs} = \frac{1}{1 + \exp\left(\frac{1}{\frac{N\_players \cdot N\_roles}{k} - f_{rs}}\right)} \quad (19)$$

$k$  is a constant value.

The values of Necessity are computed depending on the resources/roles usage. A good team is usually very well balanced, in which all players participate in the game, and contribute to the overall performance, although only a few players score goals. For this reason, it is important to have a balanced resource/role usage. In this work it means that the overall role usage should be the same for each resource. This is computed using the idea laid out in *Position of the player and role* in performance analysis, this means that not only decisions are considered, but also if this decisions were taken at the right place. This leads to reward with a higher Necessity to those roles, which resource usage is below the average and the opposite when the resource usage is above the average. (For this analysis the decision making is not taken into account when the ball gets stuck). The way the value of Necessity is computed as follows:

$$N_{rs} = \exp\left(-\delta \cdot \frac{p_r}{f_{rs}}\right) \quad (20)$$

In which  $p_r$  is the fraction of resource  $r$  usage (normalised to 1) and  $\delta$  is a constant value.

Necessity introduces some kind of random process when choosing among several roles, as it is usual in some learning process. Changes in Necessity allows agents to explore all the roles, until they start to settle on one or more roles, when they “find out” which is its best role,  $f_{rs}$  is included here to model the process of clear preference for one role. When this preference is very high, the value of  $f_{rs}$  is very high, the values of Necessity for this agent tend to be very similar, independently of the global usage, and does not affect this general preference at the revision process.

Higher Necessity values mean more chances to use a given role.

Preference of one agent over one of the resources can be modelled assigning initially higher values of Prestige, initialising  $f_{rs}$  with different values.

#### 4.4.5 Creating diversity

Diversity is one of the main aspects of [Hogg and Huberman 1991], through diversity the system is stabilised. Although diversity in this work does not play a central role in stabilisation, only under some conditions, it will be a very important point in the future.

In this case diversity will be created using different physical features for each player. In a preliminary work diversity was created developing players with different size, lineal and turning velocity and kick speed. A rule was applied to develop these players, the bigger a

player is, slower it is and a more powerful kick it has. This kind of diversity gave some interesting results:

- Players tend to roles, for which they are better fitted.
- Under extreme conditions (with a very difficult opponent) heterogeneous systems converge to roles, but only some of the homogeneous agents tend to a role.

The agents used had the following features:

Agent	Radius (cm)	Translation Velocity (m/s)	Turning Velocity (rad/s)	Kick Speed (m/s)
0	0.08	0.15	3.28	0.85
1	0.04	0.40	8.28	0.2
2	0.07	0.20	4.28	0.65
3	0.06	0.30	6.28	0.5
4	0.05	0.35	7.28	0.3

Table 1. Robot Features

These are some results:

Agent	Final Role	Dominant Population	% Final Role	% Dominant Population	Stable Roles ?
0	Defender	Defender	50	75	Yes
1	Forward	Forward	95	58	Yes
2	Goalie	Goalie	100	98	Yes
3	Leftmidfielder	Leftmidfielder	95	66	Yes
4	Righthmifielder	Righthmifielder	90	48	Yes

Table 2. Results for a diversity agents

Agent	Final Role	Dominant Population	% Final Role	% Dominant Population	Stable Roles?
0	Goalie	Goalie	100	98	Yes
1	<i>No clear role</i>	Leftmidfielder	--	60	No
2	Leftmidfielder	Leftmidfielder	90	63	Yes
3	Righthmifielder	Righthmifielder	90	63	Yes
4	<i>No clear role</i>	Defender	--	36	No

Table 3. Results for identical agents

The interpretation of each term will be explained in depth next (see example).

For this work a different kind of diversity has been introduced, although it is complementary to the first one, only this new diversity will be used for reasons of simplicity to analyse. This diversity will be based on control systems theory [Dorf 1998]. Based on experience, controllers tend to be more inaccurate when they are designed to get to the set-point as fast as possible, otherwise when it is designed to control the same system not so fast, tend to be more accurate (if both controllers are well tuned). Using this idea, diversity will be created designing players ranging from fast players, but inaccurate, to slow robots and accurate. These features are created adding white noise continuously ( $\sigma$  depending on the speed) to the set points. This simple way of creating diversity has some interesting effects on the players:

- Faster players tend to be unstable. They tend to oscillate around the set point.
- Faster players tend to show less persistence than slower ones. There rarely can accomplish they tasks when some obstacles stand on its way and tend to change their decisions.

Agent	Radius (cm)	Translation Velocity (m/s)	Turning Velocity (rad/s)	Kick Speed (m/s)
0	0.06	0.25	4.78	0.5
1	0.06	0.45	8.28	0.5
2	0.06	0.40	7.28	0.5
3	0.06	0.35	6.28	0.5
4	0.06	0.30	5.28	0.5

Table 4. Robot Features

## 4.6 Roles

Designer knowledge can be implemented in very different ways. In this example all this knowledge is built in a set of roles. Each of them has a set of high level rules associated,

***if condition1 and/or condition2 and/or..... then action1***

These rules propose actions to agents. Every action has a certainty  $\varphi$  associated ( $\varphi \in [0,1]$ ). This value depends on the level of firing of the rules (fuzzy variables) and the operators (AND/OR) leading to the action. Before a decision is made this certainties are revised by two parameters: Prestige and Necessity (see Consensus Technique). Initially each robot takes the decision with the highest revised certainty but in case there are conflicts, two agents have chosen the same action, the agent with the highest certainty wins.

Five roles have been coded using fuzzy sets. Decisions are proposed considering the position of the ball and the position of the player, which is making the decision. These five roles are:

- Goalie, its mission is to keep the goal.
- Defender, its mission is to help the goalie to keep the goal and to bring the ball towards the midfield. It never leaves its side of the field.
- Right/Left Midfielders, its mission is to pass the ball to the forward, to help the defender and be ready, next to the penalty area, in case they have a good opportunity to score a goal.
- Forward, its mission is to score goals and regain the ball at midfield, it never defends.

Each role has a number of actions associated. Some roles share one or more actions (kick\_ball, regain\_ball,.. ), some roles have their own actions (cover\_goal, right\_side, left\_side,....) .

	Goalie	Defender	Right Midfielder	Left Midfielder	Forward
Kick_ball	X	X	X	X	X
Cover_goal	X				
Defend	X				
Go_to_ball	X	X	X	X	
Attack			X	X	X
Regain_position		X	X	X	X
Go_right_side			X		
Go_left_side				X	
Attack_zone					X
Deffensive_zone		X			
Stop		X	X	X	

Table 5. Actions and Roles

One of the main problems when working with this roles, it is the lack of actions to choose among, this has created many conflicts among agents in decision making. This has led to some problems to agents when converging towards roles. This will be discussed extensively later in results.

In Annexe I more details can be found about roles and actions and how they have been codified.

#### 4.7 Consensus Technique

Certainty revision processes aim at modelling information revision processes of information coming from systems. This process can help to set high level protocols for communicating and co-operating systems. An agent can have several choices when trying to decide what to do. In this situation this agent has to select one of the choices or fusion all the information. To do that it should revise this information, depending on the confidence or preference for each resource.

In this example this process, called consensus, makes it possible, though not completely, each robot not to take conflictive actions. In order to reduce conflicts each robot has to take a different action at the same time. After this revision process, in case two agents want to take the same decision, the one with the highest certainty wins., the one that loses has to choose a different action and check if there are conflicts with other agents, this process repeats until every agent has taken a different action. The revision process is based on two parameters [de la Rosa 1993], Prestige ( $P$ ) and Necessity ( $N$ ) that modify the value of the certainty ( $\varphi$ ).

Prestige performs a linear transformation over  $\varphi$

$$\varphi' = P(\varphi) = P \cdot \varphi \quad (21)$$

Next Necessity performs a non-linear transformation over  $\varphi'$

$$\varphi'' = N(P(\varphi)) \quad (22)$$

$P, N$  and  $\varphi \in [0,1]$

## Heuristic model for revision process

$$\begin{aligned} \text{If } P \cdot \varphi \text{ greater than } \theta \text{ then } \varphi'' = N \\ \text{else } \varphi'' = P \cdot \varphi \end{aligned} \quad (23)$$

$\theta$  is a threshold, in which the value of  $N$  is included

This threshold can be approximated using a sigmoidal function. This consensus process can be implemented through different approaches, probabilistic, possibilistic and conservative. Using the probabilistic approach, the one used in this work, the resulting formula is:

$$\varphi'' = P \cdot \varphi + N \cdot \text{sigm}(P \cdot \varphi - \theta) - P \cdot \varphi \cdot N \cdot \text{sigm}(P \cdot \varphi - \theta) \quad (24)$$

Prestige can be understood as the confidence that an agent has in one role. Higher values of Prestige mean high confidence values for these roles. Prestige is a conservative parameter, if an agent has a low confidence on one role Prestige the value will very low.

Necessity is a parameter that increases  $\varphi'$  according to the necessity, which the agent has in the source of information that is being revised. A high Necessity means a high reliance on the information provided by one of the sources, this tends to increase the value of  $\varphi'$ . It is a non-conservative rule. Values depend on reliance, independently of the confidence. This non-linear feature is good for non-linear optimisation of performance indexes than the pure-linear only prestige-based techniques. These two parameters play an important role in this work. The way this parameters are applied and how they are obtained has been already explained in previous sections.

### 4.8 Implementation

This section explains how everything works together, step by step, in order to have a clear idea of how all these parts explained in depth fit together.

**Step1:** Variables assignment

- The number of own players, opponent players, features of each robot is described in *robocup.dsc*. The field inclination has to be declare in *GolfBallSim.java* , assigning a value in degrees to the variable *ANGLE*.
- Values of Necessity and Prestige are initialised, usually at 0.5. Populations values, as they are fractions, has to added up 1. Considering this constraint, agents/roles can be initialised at different values. This allows the designer to code preference of an agent(s) for one (or more) of the resources.

**Step2:** Game starts and decisions are made.

- **Position in the field:** Initially each player has a position in the field depending on their position in the file *robocup.dsc*.
- **Decision making:** Using initial values for Prestiges (normally 0.5) and Necessity (normally 0.5) each player evaluates and revised each rule for each role, in order to find which action has a higher revised certainty. Agents propose this action to their mates as its possible decision. Unless there are conflicts, two or more agents want to perform the same action, each agent transforms this decision in a lineal a turning velocity. In case two or more agents want to take the same decision the one with the highest uncertainty wins, if there is a draw the one with a lower robot identification wins (this is quite unlikely), the other agents have to take their second best choice an go through the same

process and check if there are conflicts with other agents. In order to proceed in an orderly manner, the agents with the highest uncertainty start the process, and in turn agents with lower certainties continue the process. JavaSoccer allows the user to design agents with communication capabilities, this is how this process should be done, but because the program runs very slowly using this communication method, a small change has been added using the possibility for each agent to know where its team-mates are. Having access to this information each agent can know the certainties for each of its teammates and thus which decision should be taken to avoid conflicts with teammates, based on the rules of this process. This process is repeated at each sample time in JavaSoccer ( $\approx 0.1$  seconds) for  $t$  seconds.

- **Collecting rewards:** Every agent, using its perception capability, computes rewards for each action, depending on their positions and taking into account roles.
- **Keeping track of decisions:** Every time a decision is made, it is registered together with the role and the position.
- **Communication of information:** After  $t_w$  seconds have elapsed, agents communicate to a central agent, one of the player can take over this role, rewards and decisions made through  $t_w$  seconds.
- **Reposition of players:** Each time a goal has been scored players are repositioned according to two criteria:
  1. If  $0 \leq t < t_1$  players are repositioned randomly in the initial positions
  2. If  $t \geq t_1$  players are repositioned in the initial positions according to their closer roles, in case there are conflicts the same rule as in decision making is applied.

**Step 3:** Updating Prestige and Necessity values for the next  $t_w$  seconds.

- Using information coming from each agent populations (values of  $f_{rs}$ ) are updated, using rewards for each agent and role. These changes in populations values affects all values of Prestiges for each role and agent according to equations (13-16). Equations (13-16) that model  $f_{rs}$  are non-linear differential equations. In order to approximate this function Euler method has been used.  $\Delta t$  for the approximation of each function changes through the time:

$$\begin{array}{ll}
 0 \leq t < t_1 & \Delta t = \Delta t_1 \\
 t_1 \leq t < t_2 & \Delta t = [\Delta t_1, \Delta t_2] \\
 t_2 \leq t < \infty & \Delta t = \Delta t_2
 \end{array} \tag{25}$$

Initially  $\Delta t$  is a very small value (0.01), after some time this value increases up to a certain value. This change allows the agents to go through many different situations initially until they “decide” to take over a role

- Using information from decisions and positions, role usage is computed. Using this information and normalising all values to 1, the new values of Necessity are computed.
- All population values and resource usage for each agent are saved in a file.

**Step 4:** Go to decision making in *Step 2*.

## V Experiments

### 5.1 Introduction

Some experiments will be performed to show how this method works and to analyse results and features, specially which are the interesting features and which are the deficiencies and aspects that have to be improved in the future. Two different types of team configuration will be analysed

**A:** One robot

**B:** Five different robots

Things to study:

- Stability of roles choice and convergence (A,B).
- Results coherency. Is there any relation between role and player features? (B).
- Adaptability to opponent (A,B).
- Adaptability to player changes (B)

### 5.2 Experimental Conditions

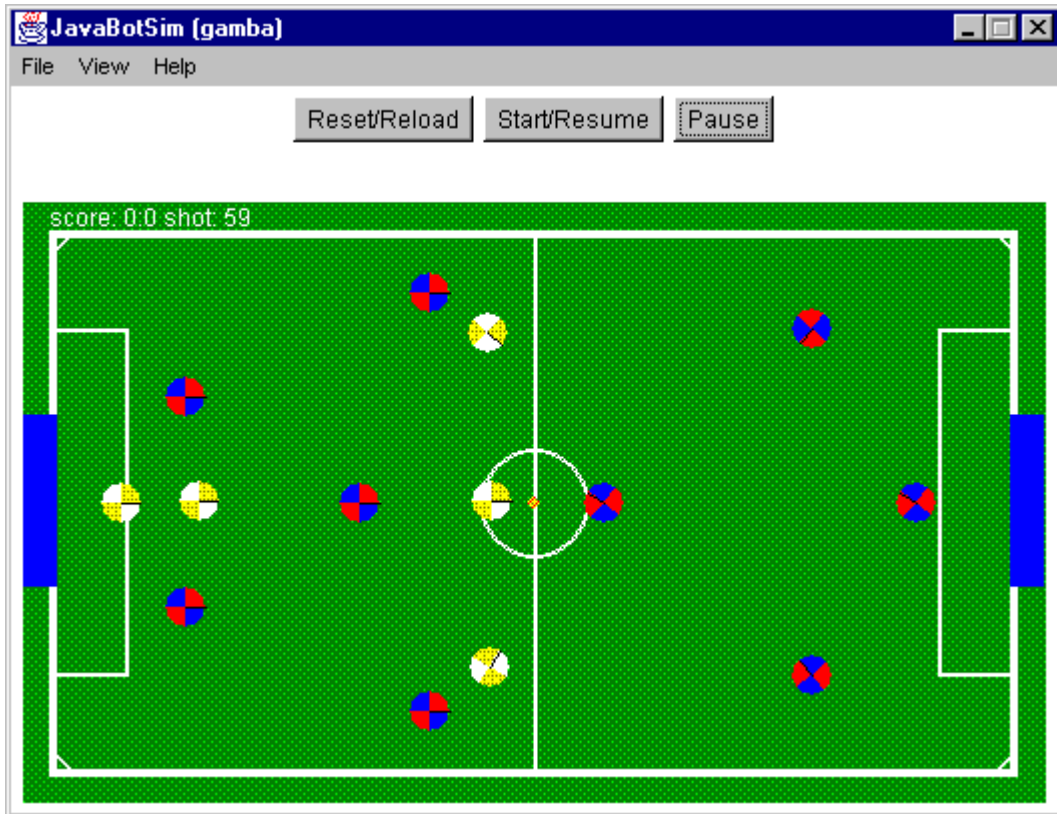
All the experiments will be performed under the same experimental conditions.

	1 Player	5 Player
Number of players	1	5
Number of roles	5	5
k	0.125	0.025
$\alpha$	1	1
$\gamma$	1	1
$\delta$	100	1
$\Delta t_1$	0.01	0.01
$\Delta t_2$	0.05	0.10
$t_1$	0	1000 seconds
$t_2$	0	1000 seconds

Table 6. Parameters Values

### 5.3 Benchmark definition

Benchmark conditions are very similar for experiments with 1 player and 5 players. This benchmark has been implemented in JavaSoccer.



**Figure 11. Standard player lay-out for 5 roles.**

Blue/red players belong to the opponent team and yellow players to the evaluated team.

Benchmark definition for 1 player

$\langle F-1, P=\{0.0174, 0.0523, 0.122\} \text{ rad}, P_r, P_o, T_e=100\text{seconds}, F_b, C, J=\{Ge-Gi\} \rangle$

In this specific benchmark origin (0,0) will be the starting point for the ball (midfield).

- $P_r$  is a pattern of evaluated robots and is the tuple  
 $\langle N_r=5 \text{ robots}, S_r=\text{round}, A_r=0.0113\text{m}^2, \langle -0.15, -0.5 \rangle \rangle$
- $P_o$  is a pattern of opponents and is the tuple  
 $\langle N_o=9 \text{ robots}, S_o=\text{round}, A_o=0.2035\text{m}^2, \langle 1.1, 0 \rangle, \langle 0.8, -0.5 \rangle, \langle 0.8, 0.5 \rangle, \langle 0.2, 0.0 \rangle, \langle -0.5, 0 \rangle, \langle -1.0, -0.3 \rangle, \langle -1.0, 0.3 \rangle, \langle -0.3, -0.6 \rangle, \langle -0.3, 0.6 \rangle \rangle$
- $F_b$  contains the features of the ball and is the tuple  
 $\langle T_b=\text{golf}, \text{CAN}=\text{rebound}, R=\langle 0.01, 0.13 \rangle, \langle 0.81, 0.43 \rangle, \langle 0.89, 0.73 \rangle, \langle 0.68, 0.34 \rangle, \langle 0.16, 0.15 \rangle, \langle 0.19, 0.42 \rangle, \langle 0.85, 0.49 \rangle, \langle 0.81, 0.46 \rangle, \langle 0.45, 0.45 \rangle, \langle 0.41, 0.90 \rangle, \langle 0.00, 0.29 \rangle, \langle 0.04, 0.69 \rangle, \langle 0.65, 0.98 \rangle, \langle 0.55, 0.40 \rangle, \langle 0.19, 0.62 \rangle, \langle 0.73, 0.37 \rangle \text{ SD}=3 \rangle$
- $C$  is the constraints and is the tuple  
 $\langle T_b=\infty, \|V_r\|=0.3, \|V_s\|=0.5, Z=() \rangle$

These experiments are performed at 1, 3 and 7 degrees.

Benchmark definition for 5 players

$\langle \mathbf{F}_b, \mathbf{P}=\{0.035,0.105,0.174\text{rad}\}, \mathbf{P}_r, \mathbf{P}_o, T_e=100\text{seconds}, \mathbf{F}_b, \mathbf{C}, \mathbf{J}=\{\text{Ge-Gi}\} \rangle$

In this specific benchmark origin (0,0) will be the starting point for the ball (midfield).

- $\mathbf{P}_r$  is a pattern of evaluated robots and is the tuple  
 $\langle N_r=5 \text{ robots}, S_r=\text{round}, A_r=0.0565\text{m}^2, \langle -0.15, 0.5 \rangle, \langle -0.15, 0.0 \rangle, \langle -0.15, -0.5 \rangle, \langle -1.0, 0 \rangle, \langle -1.2, 0 \rangle$
- $\mathbf{P}_o$  is a pattern of opponents and is the tuple  
 $\langle N_o=9 \text{ robots}, S_o=\text{round}, A_o=0.1017\text{m}^2, \langle 1.1, 0 \rangle, \langle 0.8, -0.5 \rangle, \langle 0.8, 0.5 \rangle, \langle 0.2, 0.0 \rangle, \langle -0.5, 0 \rangle, \langle -1.0, -0.3 \rangle, \langle -1.0, 0.3 \rangle, \langle -0.3, -0.6 \rangle, \langle -0.3, 0.6 \rangle$
- $\mathbf{F}_b$  contains the features of the ball and is the tuple  
 $\langle T_b=\text{golf}, \text{CAN}=\text{rebound}, \mathbf{R}=\langle 0.01, 0.13 \rangle, \langle 0.81, 0.43 \rangle, \langle 0.89, 0.73 \rangle, \langle 0.68, 0.34 \rangle, \langle 0.16, 0.15 \rangle, \langle 0.19, 0.42 \rangle, \langle 0.85, 0.49 \rangle, \langle 0.81, 0.46 \rangle, \langle 0.45, 0.45 \rangle, \langle 0.41, 0.90 \rangle, \langle 0.00, 0.29 \rangle, \langle 0.04, 0.69 \rangle, \langle 0.65, 0.98 \rangle, \langle 0.55, 0.40 \rangle, \langle 0.19, 0.62 \rangle, \langle 0.73, 0.37 \rangle \text{SD}=3 \rangle$
- $\mathbf{C}$  is the constraints and is the tuple  
 $\langle T_b=\infty, \|\mathbf{V}_r\|=0.45, \|\mathbf{V}_s\|=0.3, \mathbf{Z}=\emptyset \rangle$

## 5.4 Experiments with 1 player

In these experiments role convergence and adaptation to opponent will be analysed in depth. Three different experiments have been performed, all under the same conditions with the exception of the field inclination. Inclinations are 1, 3 and 7 degrees.  $\delta$  has been tuned in order to avoid strong oscillations. Necessity parameters tend to balance resource usage, with 1 player and 5 resources affects largely to final results. As a result of it  $\delta$  has been tuned in order to have oscillations when preference for two or more resources are very similar, otherwise clear preferences are barely affected by this parameter.

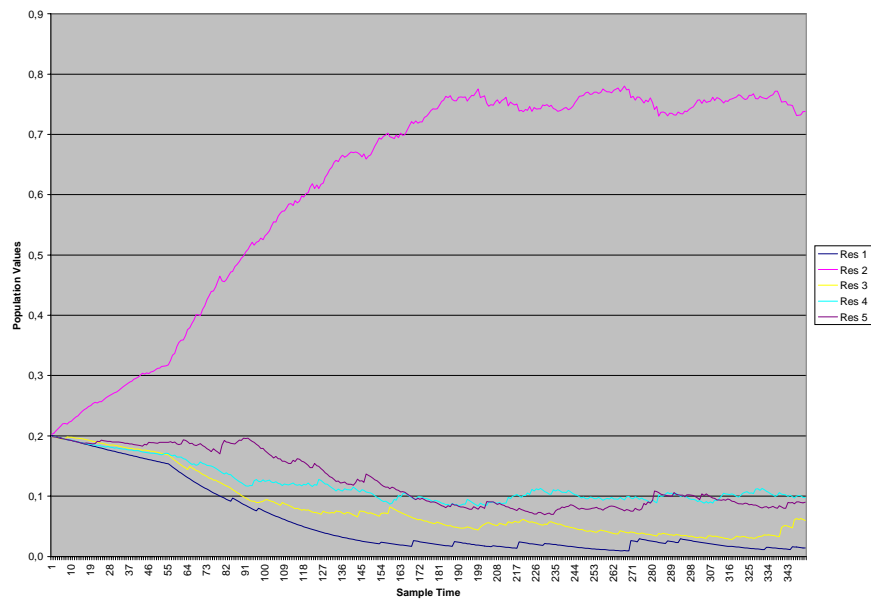
These are maximum rewards given for every action in these experiments.

	Goalkeeper	Defender	Midfielders	Forward
Set-points	max 0.005	max 0.005	max 0.005	max 0.005
Role/Position	max 0.002	max 0.002	max 0.002	max 0.002
Dribbling ability	-	-	0.1	0.55
GoalKeeper ability	4	-	-	-
Regain ball ability	4	3	1	0.5
Goal scored	-	-	1.5	2
Goal received	-2	-1.5	-	-

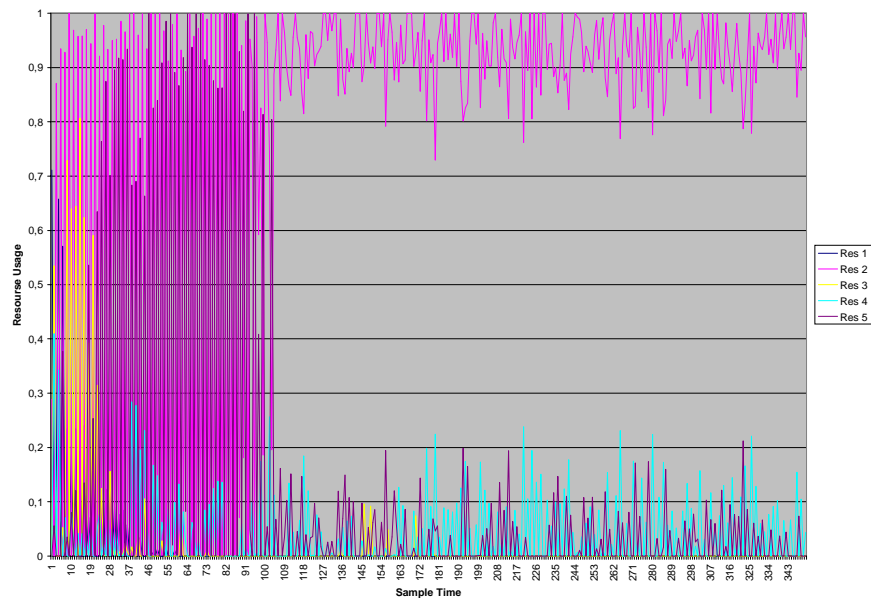
Table 7. Rewards Values

### 5.4.1 Inclination 1 degree

With this inclination the ball tend to fall down very slowly. Although the robot used for this experiment was quite slow (30 cm/s), it was able to control the ball without any problem. The results show that dominant role is forward, the player under these conditions is able to score many goals. After some transient period, the role usage settles on forward role.



**Figure 12. Population changes**



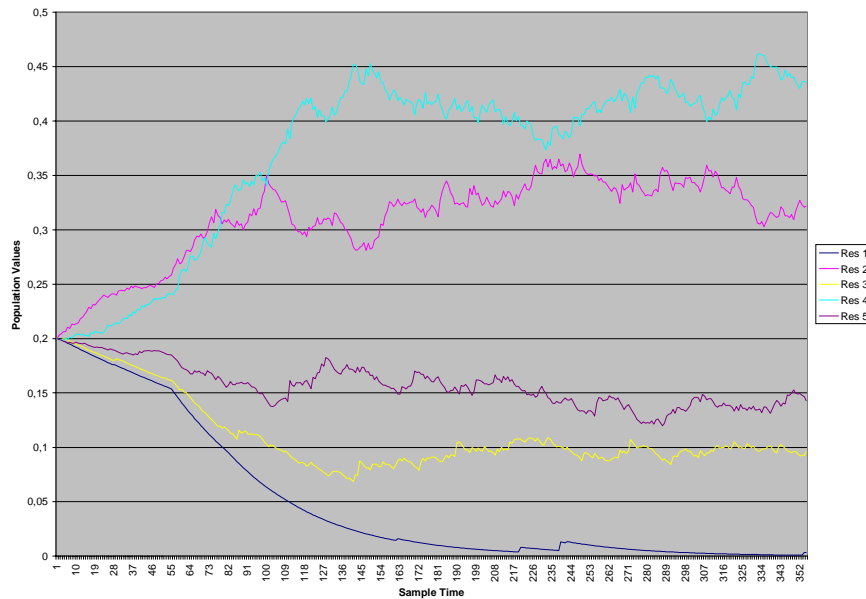
**Figure 13. Resource Usage**

Time to time other resources are used, as a result of Necessity parameter. These roles are left and right midfielders, these roles can be understood as complementary to forward, although their usage values are very small in general (around 10%). Also this secondary preference is reflected in population values.

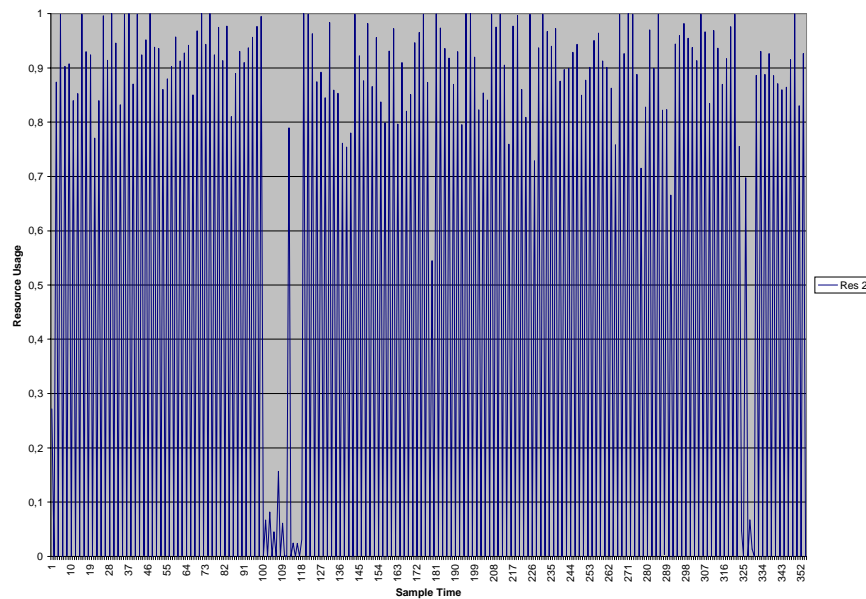
#### 5.4.2 Inclination 3 degrees

With an inclination of 3 degrees, the player is able to move forward the ball for a few seconds, with difficulties, but at the same time the ball tend to fall down quite fast. In this situation the player has to choose between two resources all the time (forward and defender). In this experiment, after several hours, the resource usage is almost identical, choosing

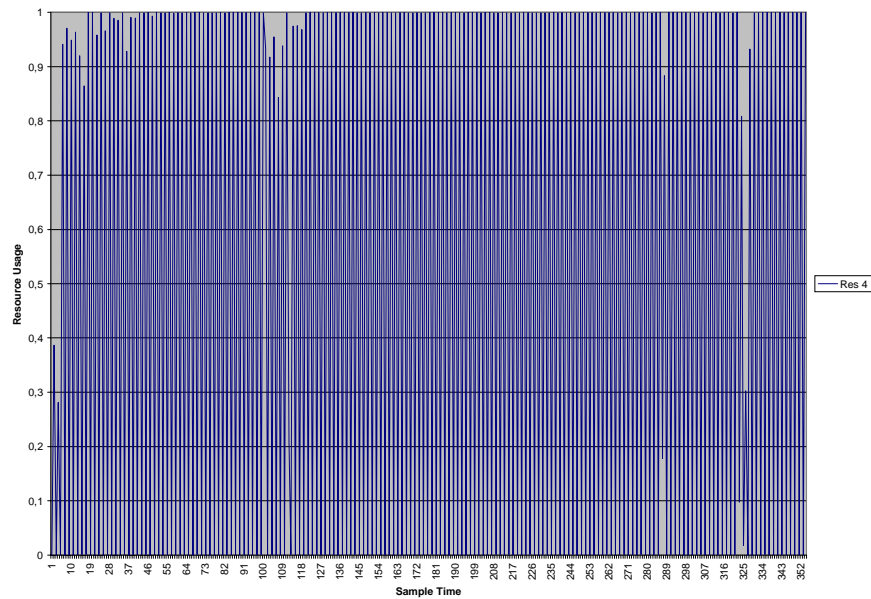
during a sample time mostly defender actions and the next sample time choosing forward actions. The reason for this indecision is the absence of a winning population. This means that prestige values are very similar and necessity values change constantly from one role to another, because of role usage.



**Figure 14. Population changes**



**Figure 15. Resource Usage for Defender**



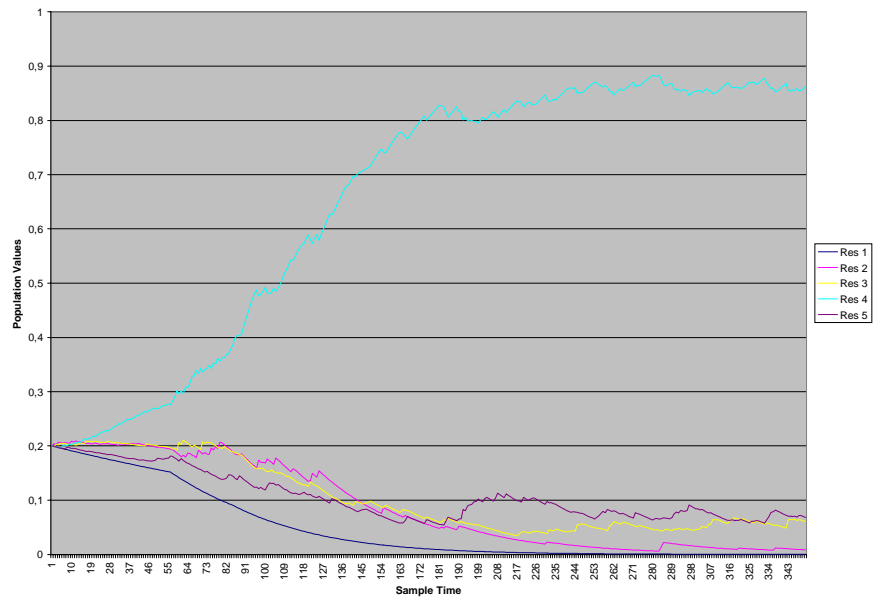
**Figure 16. Resource Usage for Forward**

Although there are two dominant resources, midfielders have also an important part of the overall populations.

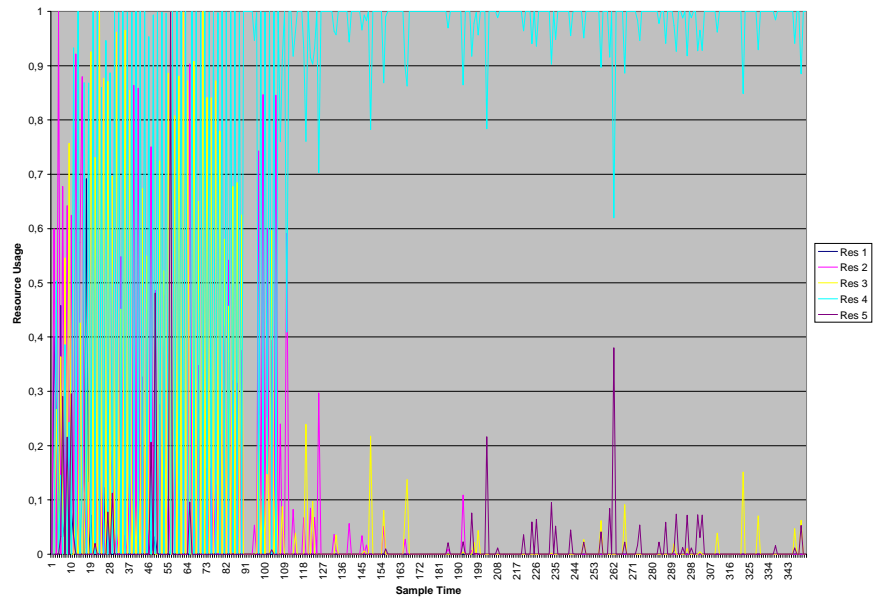
#### 5.4.3 Inclination 7 degrees

With an inclination of 7 degrees the ball tend to fall down very fast and the player have many difficulties to move the ball towards the opponent goal. This means that the dominant roles are defensive ones. In this case, and many other runs have shown that, the dominant role is defender as it can be seen in the figures. Although it seems reasonable that goalkeeper could be one of the dominant roles. It seems that there are two reasons for this apparent contradiction:

- With angle of 7 degrees the ball rolls down very fast, and many goals are scored. In this situation goalkeeper is penalised very often, leading to low rewards. Although defender can be penalised, its usual position in the filed, halfway from the goal area and midfield saves it from getting low rewards by the goals scored.
- Defender can get more rewards than a goalkeeper. While the goalkeeper saves a goal, and the ball falls down again, usually scoring a goal, a defender can gain control of the ball, for only a few seconds, and get more rewards.



**Figure 17. Population changes**



**Figure 18. Resource Usage**

As in the first experiment there is a dominant role (90%), with some usage of other roles.

### **5.5 Experiments with 5 players**

Several experiments have been performed with 5 players. These experiments aim at

- Showing how players self-organised depending on its physical abilities.
- Analysing final team configuration depending on opponent difficulty.

Although these results are preliminary, and not extensive testing has been done yet to be able compute statistical analysis, some interesting results will be shown and will be used to define future work and improvements to be done.

Experiments have been performed at different inclinations, 1,3 and 5 degrees. Three runs have been done at each inclination.

Only a table summarising final results will be shown. These results do not mean to be an extensive analysis, this belongs to future work, but representative results from the work done so far and some of the problems existing in this methodology that will lead to improvements in the future will be detailed. In this tables final states for agents will be shown. Final state is considered when the system has reached a stable configuration. Results are computed within the last 10 samples:

- **Agents:** Number of agent (there are 5 agents in overall). A table with their features is in a previous section.
- **Final Role:** Role mostly used by the corresponding agent.
- **Dominant Population:** Population (value of  $f_{rs}$ ) with the highest value.
- **% Final Role:** Role usage (%) of the most used role.
- **% Dominant Population:** (%) of the total population for the corresponding agent, of the dominant population.
- **Stable Roles:** This is stable or tends to use several roles.

Rewards have been tuned for 3 degrees, in order to have a balanced team, in which all roles tend to have similar population values. The resulting rewards are:

	Goalkeeper	Defender	Midfielders	Forward
Set-points	max 0.005	max 0.005	max 0.005	max 0.005
Role/Position	0	0	0	0
Dribbling ability	-	-	0.15	0.55
GoalKeeper ability	3	-	-	-
Regain ball ability	3	2.5	1	0.5
Goal scored	-	-	1.5	2
Goal received	-2	-1.5	-	-

Table 8. Reward Values

### Inclination 1 degree

Agent	Final Role	Dominant Population	% Final Role	% Dominant Population	Stable Roles ?
0	Goalie	Goalie	100	94	Yes
1	L-midfielder	L-midfielder	82	79	Yes
2	Forward	Forward	97	80	Yes
3	Defender	Defender	85	71	Yes
4	R-midfielder	R-midfielder	72	69	Yes

Table 9. Results experiment 1

Agent	Final Role	Dominant Population	% Final Role	% Dominant Population	Stable Roles ?
0	Goalie	Goalie	100	94	Yes
1	R-midfielder	R-midfielder	79	64	Yes
2	Goalie	Goalie	80	97	Yes
3	Forward	Forward	96	82	Yes
4	Defender	Defender	88	64	Yes

Table 10. Results experiment 2

Agent	Final Role	Dominant Population	% Final Role	% Dominant Population	Stable Roles ?
0	L-midfielder	L-midfielder	75	86	Yes
1	R-midfielder	R-midfielder	85	72	Yes
2	Forward	Forward	96	81	Yes
3	Goalie	Goalie	100	91	Yes
4	Goalie	Goalie	100	90	Yes

Table 11. Results experiment 3

**Inclination 3 degrees**

Agent	Final Role	Dominant Population	% Final Role	% Dominant Population	Stable Roles ?
0	R-midfielder	R-midfielder	92	66	Yes
1	L-midfielder	L-midfielder	90	59	Yes
2	Forward	Forward	62	45	Yes
3	Defender	Defender	33	39	Yes
4	Goalie	Goalie	100	82	Yes

Table 12. Results experiment 1

Agent	Final Role	Dominant Population	% Final Role	% Dominant Population	Stable Roles ?
0	Goalie	Goalie	99	73	Yes
1	L-midfielder	L-midfielder	93	57	Yes
2	R-midfielder	R-midfielder	94	48	Yes
3	Forward	Forward	77	52	Yes
4	Defender	Defender	36	43	Yes

Table 13. Results experiment 2

Agent	Final Role	Dominant Population	% Final Role	% Dominant Population	Stable Roles ?
0	Goalie	Goalie	80	69	Yes
1	Forward	Forward	86	52	Yes
2	L-midfielder	L-midfielder	94	62	Yes
3	R-midfielder	R-midfielder	30	40	Yes
4	R-midfielder	R-midfielder	92	71	Yes

Table 14. Results experiment 3

### Inclination 5 degrees

Agent	Final Role	Dominant Population	% Final Role	% Dominant Population	Stable Roles?
0	Goalie	Goalie	66	56	Yes
1	No clear role	Defender		41	No
2	Forward	Forward	83	72	Yes
3	L-midfielder	L-midfielder	89	58	Yes
4	R-midfielder	R-midfielder	91	58	Yes

Table 15. Results experiment 1

Agent	Final Role	Dominant Population	% Final Role	% Dominant Population	Stable Roles ?
0	R-midfielder	R-midfielder	90	72	Yes
1	Goalie	Goalie	87	64	Yes
2	L-midfielder	L-midfielder	86	57	Yes
3	Forward	Forward	82	74	Yes
4	Defender	Defender	64	64	Yes

Table 16. Results experiment 2

Agent	Final Role	Dominant Population	% Final Role	% Dominant Population	Stable Roles?
0	Goalie	Goalie	76	64	Yes
1	R-midfielder	R-midfielder	91	41	Yes
2	No clear	Defender		34	No
3	L-midfielder	L-midfielder	92	58	Yes
4	Forward	Forward	90	70	Yes

Table 17. Results experiment 3

Results show in general a self-organisation of players and roles, through an emergent process. In some examples results show do not seem to self-organised completely. In other cases robots do not always self-organise ending up with the same roles.

Results for an inclination of 1 degree show a very interesting result. Instead of having a defender and a goalkeeper, there are two goalkeepers. One possible explanation for these results is the lack of defensive situations to get rewards, most of the time players are attacking and scoring goals. In this situation most rewards for defensive roles come from set-points accomplishment, in which it is easier to get more rewards from goalie, even using the same resource two players at the same time, than from defender.

Some of the divergences observed are due to several deficiencies and some particularities:

- Roles share some actions, in some game situations, normally in defensive situations, there are not enough different actions for all the players. This leads to some players to choose other actions different from their role. This is very clear in two runs at 5 degrees. These two experiments show that defender, has a dominant population (defender), but it has no clear role when it plays with other players. This normally ends in low values of populations and having no clear role to use. Adding more roles and more actions would solve this problem.

- It does not seem that there are players specially fitted for some roles, although it exists. In the first experiments, in which other kind of diversity was used (size, speeds and kick speed), experiments showed a very clear team configuration, with some differences, at the end of the process of self-organisation.
- There is a stochastic component in collecting rewards. Rewards not only depend on own abilities, but also in the position at a given time. This method should try not to depend on this aspect, but this situation is almost impossible to deal with. This can explain, partly, diversity in results, although a general trend and a general preference can be seen, this does not mean that always the same results will be reached.

Results from overall population values are interesting to see some interesting features from heterogeneity in the system.

$\theta$	Goalkeeper	Defender	Left midfielder	Right midfielder	Forward
1°	0.16885300	0.08801007	0.12655892	0.15384927	0.46272895
3°	0.21933255	0.23363956	0.18276751	0.19423785	0.17002253
5°	0.18869478	0.30856799	0.16314221	0.18789770	0.15169733
Avg	0.19229344	0.21007254	0.15748955	0.17866161	0.26148293

Table 18. Population values for each inclination

This table shows that when playing against hard opponents (3° and 5°) defensive roles are dominant and are above the average (0.20). If the angle of inclination is 1°, the dominant role is forward. High values mean a higher preference for the resource. Although in this example there are as many roles as players, and only in goalkeeper role two players can use it at the same time as seen in the results for 1°, a global preference is manifested by this values. This results show that teams tend to defensive roles when they play against hard opponents and offensive when playing against easy opponents.

Agent	Goalkeeper	Defender	Left midfielder	Right midfielder	Forward
0	0.11509970	0.04881205	0.02666187	0.01428399	0.00801719
1	0.02315608	0.04750791	0.03325262	0.06247762	0.03096465
2	0.00641400	0.03403517	0.04542404	0.03605036	0.12732651
3	0.01173293	0.03458311	0.04294514	0.01869715	0.07480120
4	0.03589072	0.04513430	0.00920588	0.04715250	0.02037339

Table 19. Population values for each player

According to Table 19 results show preferences for roles and for each kind of agent. Agent 0 shows a clear preference for goalkeeper and also, but not so important, for defender. Agent 1 has a preference for right midfielder, although averaging the two midfielder positions, the results show a very similar value with defender. For agent 2 preference is for forward, although it should be considered that average population values for forwards are clearly above the average (0.26 vs. 0.20). For agent 3 results show also clear preference for forward and midfielder. Agent 4 is a special case and there is no clear preference for any role.

For these results, they are just preliminary experiments and do not pretend to be exhaustive only illustrative, some explanation can be inferred. Forwards are usually fast players, in this example fast players are quite unstable (fastest is agent 1), this means that it is not able to move the ball continuously and tend to lose the ball. Agents 2 and 3 are fast robots but more stable and can move the ball better. Depending on the inclination of the field players need to be faster to be able to control the ball. This reasoning can explain that agent 2 and 3 tend to be forwards, globally, and agent 1 tends to be midfielder. Agent 0 is a slow player, this

makes it not suitable for midfielder and forward roles, in general, although this depends on the opponent difficulty. Goalkeeper is a role that does not need very fast players to get rewards. This can explain agent 0 preference. Agent 4 seems that does not fit in any role and tends to resituate depending on other agent choices.

### 5.6 Changes in dynamics

An experiment has been done, in order to see what happens in the system when one of the players changes its dynamical behaviour. In this example agent acting as a forward will change its dynamics after agents has started to converge to roles. In this case this agent was agent 2. It changed its normal dynamics to 0.20 cm/sec and 4.18 rad/sec . These changes made it the slowest player in the team.

For this particular example results show how three agents change their robot adapting to the new team configuration.

As it can be seen in Figure agent 2 starts acting as forward, after a given time it experiences a change in its body (or a player substitution). Afterwards it starts playing as a defender and finally and clearly it plays as goalkeeper

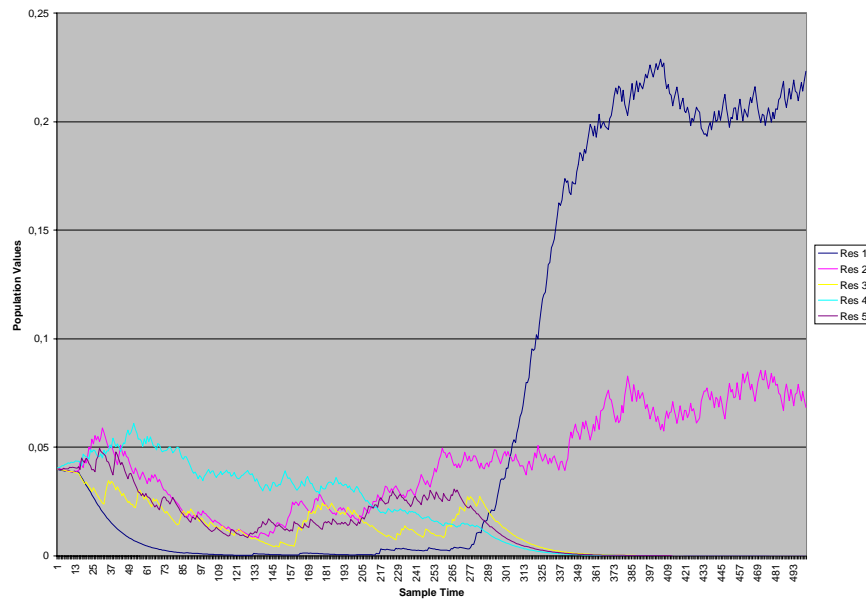
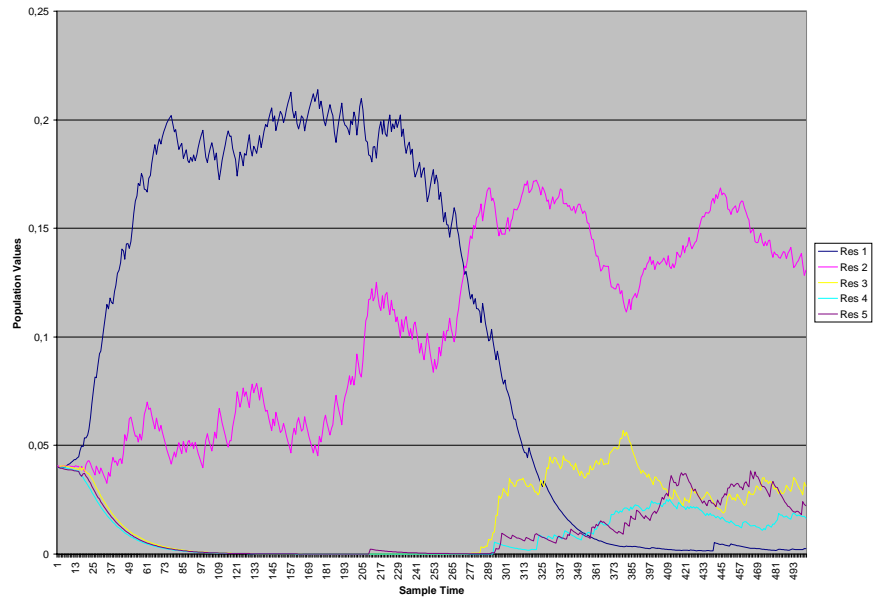
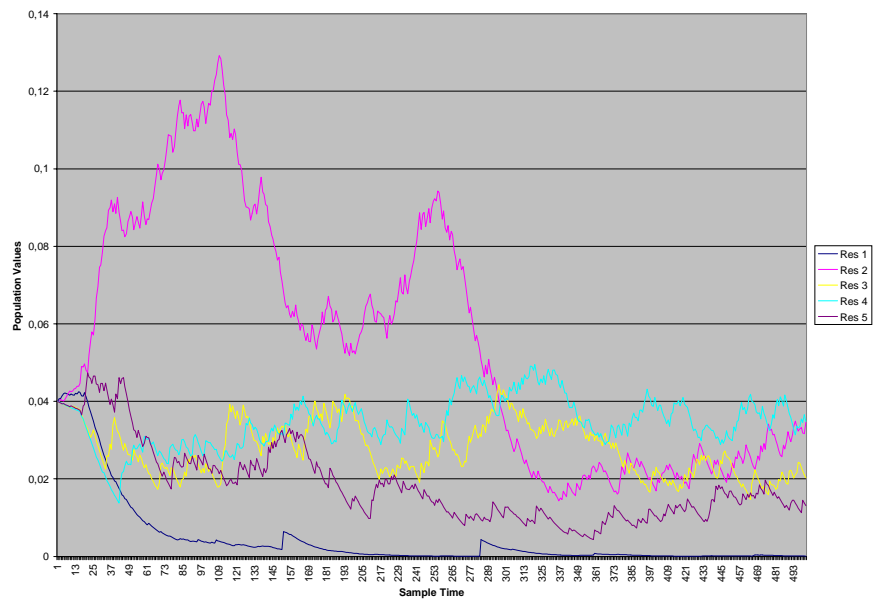


Figure 19. Population changes for agent 2

Agent 3 starts playing as goalkeeper Figure XX, afterwards agent 3 and 2 exchange their roles at the same time. The same process can be seen in agent 0, it starts as defender but ends up as forward. Agent 0 is not fit for forward role, for this reason the population values for this role do not show a clear preference for this role, although resource usage is about 70%.



**Figure 20. Population changes for agent 3**



**Figure 21. Population changes for agent 0**

From experience it is known that agent 3 is much better fitted for forward role than agent 0, although in this case agent 3 ends up as defender. As it has been inferred before the slowest agent tends to goalkeeper role.

## VI CONCLUSIONS

In this work a method has been developed, built on a previous work, to assign roles to agents depending on their physical abilities. Different physical abilities, based on dynamical behaviour concepts, have been implemented in a quite simple way that has turned out to be quite efficient. This method has proved to be quite useful, it is able to assign roles to the best suitable player, through a self-organising process. Results have shown that convergence occurs quite often, although it is not always possible and not always convergence process results in identical team organisation. This is a preliminary work and does not mean to solve entirely this problem. As it has been said this convergence process presents some deficiencies that will be solved in the future. Another interesting feature of this method is that it allows agents to adapt their team roles to the opponent difficulty, although this is also a preliminary result. The impact of heterogeneity in the process of self-organisation has been only analysed shallowly and should be a very important point for the future.

## VII FUTURE WORK

This work so far has shown some promising results, but also some problems to be tackled. There are also some issues that should be improved and better experiments should be performed. As a result of it these are the issues that in the future will be solved and analysed. Some of them are purely technical aspects and others are important research points for a thesis.

### **7.1 Technical**

#### 7.1.1 Role Design

Rules sets have been designed to allow one player for each role. This makes almost impossible to have team with several defenders, forwards... This is also a problem to have a team configuration depending not only depending on players' abilities, but also considering the opponent difficulty. Work done so far shows that this is possible when there were more roles available. For this reason in the future a complete team (11 roles for example) will be coded to be able to analyse these processes and new and more actions will be introduced in order not to have so many conflicts among different roles.

#### 7.1.2 New Knowledge Resources

All these methodology should be applied to more co-operative environments. Roles allow very little co-operation among agents, co-operation is only done to avoid taking conflictive actions. Co-operation to pass the ball or to substitute missing players, for example, has not been done so far. For this reason all these concepts of adopting decisions and behaviours depending on their physical abilities will be extended to a more complex co-operative environment.

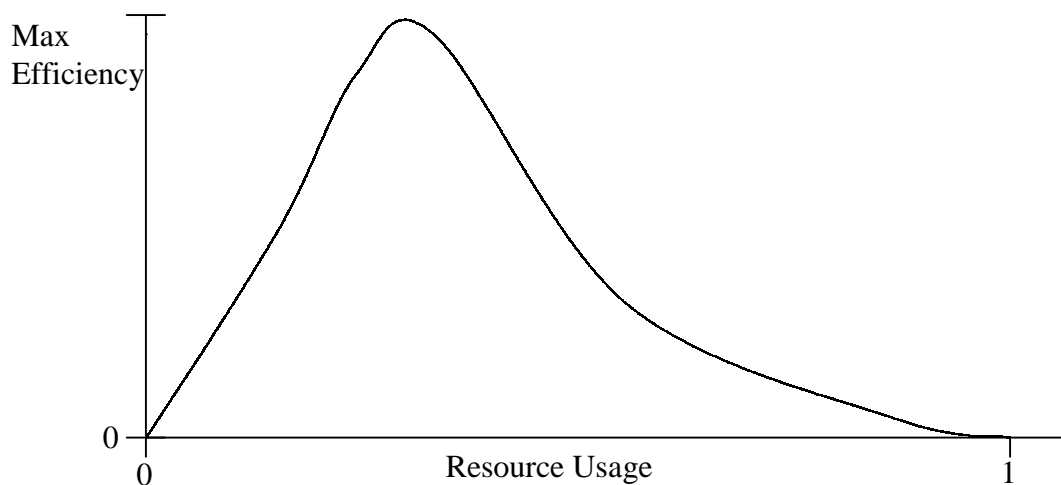
### **7.2 Research**

#### 7.2.1 Reward Assignment

During the experiments some problems and deficiencies have arisen as it has been explained in previous sections. It is extremely important to reward actions properly. In these experiments these values have been hand tuned through experience and testing. Changes in reward assignment can lead to different results under the same conditions. Although it is impossible to develop a perfect and fair way to reward agents, the way rewards are given will be considerably improved.

- One possible way is to reward roles not only taking into account the role used at that time, but also taking into account the action and the position. There are some actions that belong to different roles and some roles share some field zones, where they are designed to work. This will reward several roles at the same time and the way rewards are assigned will be much more flexible. It is quite likely that players will tend more clearly to their best fitted roles (this could eliminate stochastic aspects) and not so much divergence in results will be seen
- Also the way actions are evaluated could be improved and new aspects rewarded and features could be rewarded.

- One of the most important problems observed in reward assignment is the final configuration of the system. This problem affects especially defensive roles. Initially robot behaviour is chaotic, they do not seem to show preference over one of the resources. In this situation ball tends to fall down and most rewarded actions are defensive. If these rewards are too high the system tend to reach a defensive configuration, with defensive roles. But if these rewards are low enough system tends to a more offensive configuration, in this configuration offensive actions are more rewarded than defensive ones. To balance these two conflictive configurations a method should be found. This problem can be solved introducing a new parameter, called efficiency. Efficiency should be used to increase rewards for low global resource. This idea is partly used in the original work [Hogg and Hubermann 1991].



**Figure 22. Efficiency vs Resource Usage**

Rewards for resource usage increase up to a certain value as resource usage increases, because agents can co-operate. After a given value performance diminishes, because agents tend to have conflicts among them, although resource usage increases.

- Rewards should not be only computed using direct rewards, but also considering resource usage. Although this idea should studied in depth, to avoid extreme situations, it should allow to have a more flexible method for role assignment. Roles and action should be computed using a similar way to that presented for reward assignment to different roles at the same time.
- Some rewards are given using fuzzy sets. These methods should be extended to all rewards assignment. Zones for which roles are designed do not have clear limits. For this reasons fuzzy set is a very good way to represent these unclear limits.

### 7.2.2 Equations

The model that regulates agent relations, as it has been described, has been adapted for the purposes of this work. Some equations have been eliminated and others modified. Some equations could be better approximated to the original, especially equations (13-16) and some of them reintroduced.

### 7.2.3 Diversity Analysis

Although the impact of diversity has been only shallowly analysed, the analysis of diversity impact on team configuration should play a very important in the future work. There are many important aspects to analyse:

- How diversity affects team configuration
- How diversity affects stability and safety (perhaps reliance) in the system. Although stability can be approximated to role convergence, these future changes will affect enormously system stability, because of continuous system excitement and perhaps new parameters to evaluate that should be found.
- Differences between homogeneous teams vs. heterogeneous teams.

### 7.2.4 Teams Performance Analysis and Optimal Solution

The process of player to roles allocation tend to converge, it is important to know whether this final team configuration is optimal, and how team performance can be evaluated, difference of goals? . Perhaps this is not enough in this problem, in which abilities are rewarded, but not team strategy. This is going to play a very important role in future work .

## VIII BIBLIOGRAPHY AND REFERENCES

### **8.1 Main Bibliography**

[Ali 1999] Khaled Subhi Ali, MultiAgent Telerobotics, Matching Systems to Tasks, PhD dissertation, Georgia Institute of Technology, June 1999.

[Asada, Kuniyoshi et al 1997] Asada M., Kuniyoshi Y., et al. The RoboCup Physical Agent Challenge, First RoboCup Workshop in the XV IJCAI-97 International Joint Conference on Artificial Intelligence, pp.51-56, 1997.

[Balch 1997] Balch T, JavaSoccer, RoboCup-97: Robot Soccer World Cup I in Lecture Notes in Computer Science, pp 181-187, 1997.

[Balch 1998] Balch T, Behavioural Diversity in Learning Robot Teams, PhD Thesis, Georgia Institute of Technology, December 1998.

[Barman, Kingdon et al 1993] Barman R. A., Kingdon S. J et al, Dynamite: A Testbed for Multiple Mobile Robots, Proceedings of the IJCAI-93 Workshop on Dynamically Interacting Robots, 1993.

[Bonabeau, Theraulaz et al 1997a] Bonabeau E., Theraulaz J-L. et al, Self-Organisation in Social Insects, Trends Ecological Evolution. 12, 183-193, 1997.

[Bonabeau, Theraulaz et al 1997b] Bonabeau E., Theraulaz J-L. et al, Adaptive Task Allocation Inspired by a model of Division of Labour in Social Insects, , Bio Computation and Emergent Computing, pp 36-45, 1997.

[Bonabeau, Theraulaz et al 1999] Bonabeau E., Theraulaz J-L. et al, Dominance orders in animal societies: the self-organisation revisited, Bull. Math. Bio.,1999.

[Brooks 1989] Brooks, R. A., How To Build Complete Creatures Rather Than Isolated Cognitive Simulators, Architectures for Intelligence, K. VanLehn (ed), Erlbaum, Hillsdale, NJ, pp. 225--239. Fall 1989.

[Brooks 1990a] Brooks, R. A., Challenges for Complete Creature Architectures, First International Conference on Simulation of Adaptive Behaviour, pp. 434-443, Paris, France, September 1990,

[Brooks 1990b] Brooks, R. A., Elephants Don't Play Chess,, Robotics and Autonomous Systems Vol. 6, pp. 315, 1990.

[Brooks 1991] Brooks, R. A., New Approaches to Robotics, Science, Vol. 253, pp.1227-1232, September 1991.

[Caldarelli, Higgs and McKane 1998] Caldarelli G., Higgs P. G. and McKane A. J., Modelling Coevolution in Multispecies Communities, Journal of Theoretical. Biology. 193, pp345-358, 1998.

[Chatterjee and Chatterjee 1987] Chatterjee S. and Chatterjee S., On Combining Expert Opinions, American Journal of Mathematical and Management Sciences, 1987.

- [Devine and Paton 1997] Devine P., Paton R. C., Herby: An Evolutionary Artificial Ecology, ICEC 1997.
- [Devine, Paton and Amos 1997] Devine P., Paton R. and Amos M., Adaptation of Evolutionary Agents in Computational Ecologies, BCEC 97, Sweden., 1997.
- [Dorf 1998] Dorf R.C., Modern Control Systems, Addison-Wesley, Ed. 8, 1998.
- [Ekbom and Astor 1996] Ekbom K. and Astor E., Investigations of Multi-Agent Systems in Computational Markets and Ecosystems, Research-Report 14-96, University of Karlskrona, October 1996.
- [Eoyang and Conway 1999] Eoyang G., Conway D. J., Conditions that Support Self-Organisation in A Complex Adaptative System, International Association of Facilitators, Annual Meeting, January 1999.
- [Gerkey 1998] Gerkey P.G., Task Allocation for Heterogeneous Robots, Thesis for the Bachelor of Science, Tulane University, April 1998
- [Gerkey and Mataric 2000] Brian P. G. and Mataric M.M, Murdoch: Publish/Subscribe Task Allocation Heterogeneous Agents, Proceedings of Agents 2000. pp 203-204, June 2000.
- [Greenwald and Kephart 1999] Greenwald A.R. and Kephart J.O., Shopbots and Pricebots, XVI IJCAI-99 International Joint Conference on Artificial Intelligence, Vol. 1, pp.506-511, Stockholm, August 1999.
- [Grabowski, Navarro-Serment et al 2000] Grabowski R., Navarro-Serment Luis E. et al, Heterogeneous Teams of Modular Robots for Mapping and Exploration, 2000
- [Hong and Page 1997] Hong L., Page S.E., Problem Solving by heterogeneous Agents, Submitted for publication in 1997, University of Iowa.
- [Hogg and Huberman 1991] Hogg T. and Huberman B. A., Controlling Chaos in Distributed Systems, IEEE Transactions on Systems, Man, and Cybernetics, Vol. 21, No. 6, November/December 1991.
- [Hraber, Jones and Forrest 1997] Hraber P.T., Jones T., Forrest S., The Ecology of Echo, Artificial Life, Volume 3, Number 3, 1997.
- [Johnson, de la Rosa and Kim 1998a] Johnson J., de la Rosa J.Ll., and Kim J.H., "Benchmark Tests in the Science of Robot Football" Proceedings IEEE of Miroso-98, pp. 85-89, Paris 1998. R. J. Stonier (ed). Univ. Central Queensland.
- [Johnson, de la Rosa and Kim 1998b] Johnson J., de la Rosa J.Ll., and Kim J.H., "Benchmark Tests of Robot Soccer Ball Control Skills" Proceedings IEEE of Miroso-98, pp. 91-93, Paris 1998. R. J. Stonier (ed). Univ. Central Queensland.
- [Kaelbling, Littman and Moore 1996] Kaelbling L.P., Littman L.M and Moore A.W, Reinforcement Learning: A Survey, Journal of Artificial Intelligence Research, pp 237-285, May 1996.
- [Kauffmann 1991] Kauffmann S.A, Antichaos and Adaptation, Scientific American, pp. 78-84, August 1991.

- [Kitano, Veloso et al 1997] Kitano H., Veloso M., et al., The RoboCup Synthetic AgentChallenge 97, XV IJCAI-97 International Joint Conference on Artificial Intelligence, Vol 1, pp.24-29, Nagoya, August 1997.
- [Kube and Zhang 1992] Kube C.R, Zhang H., Collective Robotic Intelligence, Second International Conference on Simulation of Adaptative Behaviour, pp 460-468, 1992.
- [Lander 1994] Lander S.E., Distributed Search and Conflict Management Among Reusable Heterogeneous Agents, PhD Thesis, University of Massachusetts, May 1994.
- [Mackworth 1999] Mackworth A, The Dynamics of Intelligence: Constraint-Satisfying Hybrid Systems for Perceptual Agents, AAAI Spring Symposium in Hybrid Systems and AI, 1999.
- [Mataric 1994] Mataric M.J, Interaction and Intelligent Behaviour, PhD dissertation, MIT, May 1994.
- [Miller and Drexler 1988] Miller M. S. and Drexler K. E., Markets and Computation: Agoric Open Systems, The Ecology of Computation, B. A. Huberman, Ed Amsterdam: North-Holland, pp 133-176, 1988.
- [Mitchell 2000] Mitchell M., Life and Evolution in Computers, Darwinian Evolution Across the Disciplines, 2000.
- [Nagendra, Lesser and Lander 1995] M V Nagendra Prasad, Victor R Lesser and Susan E Lander "Learning Experiments in a Heterogeneous Multi-agent System", IJCAI-95 Workshop on Adaptation and Learning in Multiagent Systems, Montreal, Canada, August, 1995.
- [Nagendra, Lesser and Lander, 1996] M V Nagendra Prasad, Victor R. Lesser, and Susan E. Lander. Learning organisational roles in a heterogeneous multi-agent system. In Adaptation, Coevolution and Learning in Multiagent Systems: Papers from the 1996 AAAI Spring Symposium, pages 72-77, Menlo Park, CA, March 1996. AAAI Press.
- [Oller, de la Rosa , del Acebo 1999] Oller A., de la Rosa J. Ll., del Acebo E., DPA: Architecture for Co-operative Dynamical Physical Agents, MAMA AW'99, June 1999.
- [Parker 1994] Parker L.E., Heterogeneous Multi-Robot Co-operation, PhD Thesis M.I.T, May 1994.
- [Parker 1997] Parker L.E., Co-operative Multi-Robot Observation of Multiple Moving Targets , Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '97), pp. 1591-1598 1997.
- [Parker 1998] Parker L.E., Toward the automated synthesis of co-operative mobile robot teams, Proceedings of SPIE Mobile Robots XIII, vol. 3525, pp 82-93, 1998.
- [Parker 1999] Parker L.E., A Case Study for Life-Long Learning and Adaptation in Co-operative Robots Teams, Proceedings of SPIE Sensor Fusion and Decentralised Control in Robotic Systems II, vol. 3839, pp. 92-101, 1999.
- [Reynolds 1987] Reynolds C.W., Flocks, Herdes and Schools: A Distributed Behavioural Model, Computer Graphics, vol 21, n<sup>o</sup>4, pp 25-34, July 1987.

- [de la Rosa 1993] de la Rosa J. Ll., Heuristics for Co-operation of Expert Systems, Application to Process Control, PhD Thesis, Universitat Autònoma de Barcelona (UAB), 1993.
- [de la Rosa 1997] de la Rosa J. Ll., Oller A., et al., Soccer Team based on Agent-Oriented Programming, Robotics and Autonomous Systems. Ed.Elsevier, Vol 21, pp.167-176, October 1997.
- [de la Rosa J. Ll, García R, Innocenti B., et al 1999] de la Rosa J. Ll., García R., Innocenti B., et al., Rogi Team Real: Research on Physical Agents, 3rd Workshop on RoboCup, 16th IJCAI, July 31-August 6, 1999.
- [de la Rosa, Innocenti et al 2000] de la Rosa J. Ll., Innocenti B., et al, An example of Dynamical Agents, Workshop in EuroRobocup 2000, Amsterdam, June 2000.
- [de la Rosa, Duhot and Muñoz 2000] de la Rosa J.Ll., Duhaut D., Muñoz I., WAC 2000 , Tarragona, September 2000.
- [Rocha 1999] Rocha L. M., Syntactic Autonomy or Why There is no Autonomy without Symbols and how Self-Organisation Systems Might Evolve Them, New York Academy of Sciences, 1999.
- [Stone 1998] Stone P., Layered Learning in Multi-Agent Systems, PhD Thesis, Carnegie Mellon University, December 1998.
- [Shoham 1993] Y. Shoham. "Agent-oriented programming". Artificial Intelligence, 60:51-92, 1993.
- [Torra 1998] Torra V., On Considering Constraints of Different Importance in Fuzzy Constraint Satisfaction Problems, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, Vol 6, No 5, pp 489-501, 1998.
- [Unsal 1993] Unsal c, Master Thesis: Self-Organisation in Large Populations of Mobile Robots, Virginia Polytechnic Institute, May 1993.
- [Unsal 1997] Unsal C., Intelligent Navigation of Autonomous Vehicles in an Automated Highway System: Learning Methods and Interacting Vehicles Approach, PhD Thesis, Virginia Polytechnic Institute, 1997.
- [Youssefmir, Huberman and Hogg, 1998] Michael Youssefmir and Bernardo A. Huberman and Tad Hogg. Bubbles and Market Crashes, Computational Economics, V12, 1998.
- [Wooldridge and Jennings 1994] Wooldridge, M. Jennings, M. "Intelligent Agents: Theory and Practice". Knowledge Engineering Review. 1994.
- [Wooldridge 1999] Wooldridge, M. "Intelligent Agents". Multiagent Systems. Editor G. Weiss, The MIT Press, April 1999.
- [Zhang and Mackworth 1994] Zhang Y. And Mackworth A.K, Specification and verification of constraint-based dynamic systems, Principles and Practice of Constraint Programming, Lecture Notes in Computer Science number 874, pp 229-242, 1994.
- [Zhang and Mackworth 1995] Zhang Y. And Mackworth A.K, Constraint Nets: A Semantic Model for Hybrid Dynamic Systems, Theoretical Computer Science 130, pp 211-239, 1995.

[Zhang and Mackworth 1998] Zhang Y. And Mackworth A.K, A Multi-level Constraint-based Controller for the Dynamo98 Robot Soccer Team, Proceedings of the second RoboCup Workshop, pp 353-359, 1998.

## **8.2 Complementary Bibliography**

[Clearwater and Huberman 1993] Clearwater S.H. and Huberman B., Thermal Markets for Controlling Building Environments, Technical Report SPL-93-050, P93-00103, Systems & Practices Laboratory, Palo Alto Research Center, Palo Alto, California, 1993.

[Shanon 1949] Shanon C.E., The Mathematical Theory of Communication, University of Illinois Press, 1949.

[Simon 1969] Simon H. A., The Sciences of the Artificial, MIT Press, 1969.

[Waldspurger, Hogg et al 1992] Waldspurger C.A., Hogg T. et al, Spawn: A Distributed Computational Economy, IEEE Transactions on Software Engineering, Vol.18 No2, pp. 103-117, Feb. 1992.

[Yager 1988] Yager R. R., On ordered weighted averaging aggregation operators in multi-criteria decision making, IEEE Transactions on Systems, Man and Cybernetics, 1, pp 183-190, 1988.

[Yovits, Jacobi and Goldstein 1962] Yovits M.C, Jacobi G.T., Goldstein G.D., Self-Organizing Systems, Self-Organising Systems, Washington D.C., McGregor&Werner, 1962.

[Zadeh 1965] Zadeh L.A, Fuzzy sets, Information an Control, vol, pp. 338-353, 1965.