25

A Control Agent Architecture for Cooperative Robotic Tasks

Enrique González[1], Fernando De la Rosa[2], Alvaro Sebastián Miranda[1], Julián Angel[2] and Juan Sebastián Figueredo[1]
[1]*Pontificia Universidad Javeriana*
[2]*Universidad de los Andes*
Colombia

1. Introduction

In Robotics a multi-robot approach is mandatory when the performance, robustness needed or functionality cannot be fulfil with only one robot. In this context, cooperation is a very important aspect to be taken into account because it allows that a set of autonomous robots to achieve the task by adding their skills and resources. A natural approach to accomplish a multi-robot task is by decomposing it into cooperative actions, each one executed by a group of the robots where every robot takes a well defined role. To have an intentional cooperation level working with a Multi-Robot System (MRS), explicit mechanisms and control architectures have to be defined. In order to analyse or design a multi-robot system, it can be viewed as a Multi-Agent System (MAS) composed of physical agents. Many aspects have influence in the design of these complex systems in robotics: task decomposition, task allocation, role assignment, inter-agent interference and competition, agent cooperation, coordination, conflict resolution, negotiation and inter-agent communication.
Cooperation in the context of MAS has emerged to provide better use and performance of the agents and their capabilities. Following the approach proposed by Ferber (Ferber, 1999), cooperation can be seen as the conjunction of three components:
Collaboration: it is centered in task allocation. In order to assign which agent has to accomplish a specialized task, interaction protocols can be used. These dialogues allow to take into account not only the capabilities of the agents, but also their availability.
Coordination: it deals with the synchronization and planning issues. For this, it is necessary to determine at what time an agent should perform a task. The global team performance depends on providing good timing to each agent.
Conflict Resolution: usually agents share resources in a concurrent way, which easily can produce conflicts and even dead locks; thus it is necessary to incorporate mechanisms to prevent, avoid or solve conflicts.
Normally, these three components are obtained by the use of interaction protocols that define well structured dialogues between the cooperative agents. In just one sentence, cooperation is achieved by using collaboration, coordination and conflict resolution mechanisms supported by structured communications between agents.
In order to achieve cooperation, different architectures and techniques have been proposed, as shown in section 2. In this chapter, the Multi-Resolution Cooperation Control (MRCC)

approach is presented, which proposes a control architecture for intentional cooperation distributing responsibilities on multiple levels by applying the multi-resolution principle. This principle implies a hierarchical decomposition of the MAS cooperative control, where each layer manages the decisions at different granularity and abstraction levels. The layers of decomposition are: the system level (the higher level) which determines the global strategy and affects the lower levels trying to cope with the team goals and to obtain a better global system performance. Then, the formation level, which aims to give structure to the team, defines zones and assigns structural roles. Next, the micro-social level creates agent societies charged of executing cooperative actions; each agent in a micro-society assumes one of the required cooperative roles. Finally, the agent level (the lower level) includes the individual control of agents acting to achieve role goals. The proposed control architecture is validated by means of a functional prototype developed to play robotic soccer. This experimental context is characterized by a high dynamic environment, with multiple situations and possible game actions where the robotic team cooperation and its performance are critical.

This chapter is organized in three parts described as follows. In the first part, some of the most relevant previous works about control architectures in multi-robot systems is presented; on one hand, the idea is to show how different approaches consider and combine the aspects involved in the design of multi-robot systems, and on the other hand, to show their application to specific problems. The next part, which includes several sections, explains our approach of control architecture MRCC for intentional robotics cooperation, the micro-social level is analyzed in detail; it also introduces the principal considerations of the validation of the approach in the context of robotic soccer. The chapter ends with the presentation of some results focused in the cooperative action mechanism and a final discussion.

2. Related work

In this section, a selection of outstanding works, which propose control architectures in mobile robotics, is presented. These works highlight the main problems and considerations in the design and implementation of these architectures in order to accomplish cooperative tasks with a group of autonomous mobile robots.

ACTRESS (acronym of ACTor-based Robot and Equipments Synthetic System) is a general architecture for robotics systems which are composed of multiple robots and other teams/components (Asama et al., 1989). Each element in the system is represented by means of the concept of *robotor* (robotic actor). A *robotor* has associated data, capabilities of processing, making decisions, movement, manipulation and message sending. A robotics system is composed of *robotors* with different structure and functionality which use a communication protocol in order to achieve the coordination among them. However, this work does not precise/define a mechanism for solving cooperative tasks. It presents experimental results of a system composed of two physical micromouses and one virtual micromouse executing an obstacle pushing task.

Mataric presents a multi-robot architecture based on (basic) behaviors (Mataric, 1995). The author affirms that, for each domain/task, it is possible to find a minimal set of behaviors which are necessary for achieving its objective. The combination of these behaviors in a robot (with concurrent execution or by exclusive selection) allows to generate more complex behaviors. Each behavior has associated a small set of activation rules (in some cases, only

one rule). The collective behavior is the result of local interactions between robots when executing the basic behaviors in each robot. The main purpose is maximizing the synergy among multiple robots in order to achieve the objective of the task, minimizing the inter-agent interference. For the task, it is used a set of 20 mobile robots equipped with infrared and touch sensors, with a broadcast communication mechanism, limited to a distance. The behavior-based architecture was tested experimentally, evaluating the flocking collective behavior by concurrent execution of the basic behaviors avoidance, aggregation and wandering. Independently, the foraging collective behavior was evaluating by selective execution based on the basic behaviors avoidance, dispersion, following, homing, wandering, grasping and dropping.

ALLIANCE (Parker, 1998) is an architecture of distributed control and based on behaviors which supports fault tolerance, trusted and adaptable to small or medium team of heterogeneous robots (with uncertainty in their actions). The agents/robots have the capability of making decisions in an autonomous way according to the task to solving and the actions of the other robots. The architecture allows to solve missions composed of independent tasks in an adequate order in dynamic environments. Each agent has multiple sets of behaviors (competences), each set designed for solving a high level task. The activation of each set depends on the evaluation of a motivational behavior/model getting a positive numeric value. Whether this value is greater than a threshold value, the set of associated behaviors is activated. In this case, the rest of the sets of behaviors are inhibited. This evaluation is continually done in each robot and allows to adapt its behavior according to environmental changes, communication faults or fault(s) in any robot. The low level behaviours/abilities (e.g. collision avoidance) are always active. A robot periodically informs to the rest of robots, the task that it is solving/executing. The motivational behavior/model takes into account the sensorial information of the robot, the inter-robot communication, the inhibition of the other behaviors and the internal motivations. The internal motivations correspond to the *impatience* and to the *acquiescence*. The impatience allows a robot to deal with situations in which other robots fail. The acquiescence allows to deal with situations in which a robot decides to give up because it could not complete them in a determined time. In this way, a robot searches to participate in tasks where it could be more productive. The architecture does not provide an explicit coordination mechanism among the team members, in the case that it will be necessary. This architecture was tested experimentally in a hazardous waste cleanup mission with a team of 3 homogeneous robots, each robot equipped with infrared sensors, touch sensors and a gripper. For the mission, four independent tasks were identified: find-locations-methodical, find-locations-wander, move-spill(loc), and report-progress. In addition, the avoidance-obstacle is included as a low level behavior. There is a report of successful tests of the mission under different functioning conditions of the robots.

Simmons et al. propose a multi-robot system (MRS) for tasks where, an heterogeneous robot team and a mechanism of explicit coordination among the robots, are necessary in order to achieve the objective (Simmons et al., 2001)(Hershberger et al., 2002). The proposal is based on a layered architecture belonging to each robot. This architecture is composed of: (a) a superior planning layer for achieving high level objectives; (b) an executive intermediate layer for synchronizing agents, tasks sequences and monitoring the execution; and (c) a behavioral layer which is connected/related with sensors and actuators. There are connections between the layers robot's and also there are multi-robot connections. The architecture supports dynamic formations of the team. Each agent dynamically generates a task sub-tree. The decomposition

of a task between the MRS, the time restrictions and the sub-objectives restrictions in the MRS are defined by using the Task Description Language (TDL). Each agent executes its role by using its task tree. A foreman agent facilitates the execution of cooperative actions constituting the group of robots and assigning their roles. Once the roles have been assigned, the foreman agent monitors and coordinates the robot actions during a cooperative action by means of explicit communication. The architecture was experimentally tested in the execution of a large-scale assembly, using 3 autonomous heterogeneous robots (a 6 DOF Robocrane, a roving eye and a 5 DOF mobile manipulator).

CS Freiburg (Weigel et al., 2002) is a winner team of middle-size robot league (F2000) of Robocup in 1998, 2000 and 2001 and, third place in 1999. The player's (local) perception is obtained by using a LRF Sick LMS200 sensor and a digital video camera. By using the LFR sensor information, each robot has the ability of auto-location in the play field. Additionally, filtering the information of the play field borders, it is possible to detect the other players in the LRF's field of view. The camera allows to detect the ball. Each robot resends its location estimation, the detection of other players and the estimation of the ball to a central processing node. This node uses a sensorial fusion model in order to obtain an estimation of players' location (distinguished by team) and of the ball (active elements). The team strategy is based on the concept of formation which defines positions of the robots in specific areas of the play field. The goalkeeper robot has a well-defined role. The three other robots have dynamic roles: a robot with the active role (related with a direct action on the ball), other has the strategic role (supports the defense) and the last one has the role of support (support to the defense or pass receiver). Each role additionally has a priority. The central node dynamically defines the formation and the roles assignment which communicates to the players. Each robot works in an autonomous way by using local information and information about the active elements (if it is available from the central node). The robot evaluates according to its role, the position most appropriated (preferred pose). Each robot does the path planning in order to achieve its appropriated position which is communicated to its team in order to avoid collisions. The solution of highest priority is the one of the active role. A player continually evaluates its estimation in order to carry out each possible role, taking into account its priority and communicates it to its team. In the case of possible conflicts for a role, there is a negotiation mechanism. Each player has an emergency strategy which assigns the preferred area in the play field, in case of the existence of communication problems; in this way, the team covers all the most important zones.

Lima and Custódio propose an architecture composed of 3 types of behaviors available in a team of robots: (a) organizational behaviors related to the roles in a team, (b) relational behaviors which depend on the relations in a team, and (c) individual behaviors related to individual abilities (Lima & Custódio, 2005). A behavior is based on the concept of operator. An operator implements actions which produce an individual or team behavior. A team behavior implies the establishment of commitments among participant agents; this establishment implies a communication process among the operators applied among the participant agents. The architecture considers 3 levels: (1) Organization team level where a strategy is defined for the team and its objective. A strategy consists in activating a subset of behaviors for each agent according its role. (2) Task coordination level where the sequence of appropriated behaviors is selected for executing the strategy and accomplishing the team objective. This level is in charge of the coordination of individual and team behaviors. The coordination uses mechanisms of event detection, validation of commitments and synchronization messages among the participants. (3) Behavior execution level where the

basic/primitive functions are executed, taking into account the sensors and the actuators of each agent/robot. The implementation of a behavior is based on a finite state automaton, which is executed in one or multiple robots. An implementation of this architecture was done for robotic soccer where the pass is used as a team behavior. However, this architecture has limitations in tasks with strong coordination (coupling) among participants (e.g. team formations).

ETHNOS (acronym of Expert Tribe in a Hybrid Network Operating System) is a framework which allows to design distributed robotic applications (Sgorbissa, 2006). ETHNOS is composed of a set of services and characteristics that suit some tasks of mobile robotics: a distributed real-time operating system, a dedicated network communication protocol designed for both single robot and multi-robot systems; an object oriented Application Programming Interface (API) based on the C++ language. The control in ETHNOS is based on the concept of concurrent agents and experts in specific activities. These agents are process with real time restrictions and with priorities which can be distributed in a computer network. ETHNOS allows to execute these agents/robots with different conceptual solutions but they can integrate explicit communication and coordination capabilities, necessary in a multi-robot system. In the case of robotic soccer, the coordination in ETHNOS is based on the formation concept which has associated a set of roles, a role for each robot player which includes its activities/responsibilities and its position in the play field. The assignment of roles is dynamic according to the actual situation and the capabilities of each robot. This framework was tested by the Azzurra Robot Team composed of 5 different robot models, each one with its own control logics. This team won the second place in the middle-size robot league of Robocup (1999).

An architecture based on behaviors is the one proposed for a Multi-Robot System (MRS) in dynamic and unknown scenarios (Quiñonez et al., 2009). The behaviors are defined for a pursuit-evasion (surveillance) task between two robot teams. According to its objective, each team has associated a set of behaviors. These behaviors are represented in a state machine automaton. The autonomy and the decision making process of each robot are based on this automaton. The behaviors for each team are of two types: Navigation Behaviors (defining the movement possibilities) which consist of behaviors: searching robot, avoiding obstacle, unblocking, following robot and avoiding robot. The second type is the Communication Behaviors (by using an indirect mechanism based on traces/marks left by the "evaders") defined by the behaviors: releasing puck, following puck and avoiding puck. Each behavior is modeled by an artificial neural network and implemented by using evolutionary algorithms. There are some test results in simulation with 3 pursuers and 1 evader.

De la Rosa and Jimenez evaluate four multi-robots architectures based on behaviors with different levels of knowledge, coordination and organization of the robots (De la Rosa & Jimenez, 2009). Each of the architectures has a central process with information of the robots and of the scenario map. The relation of its behaviors is defined with a finite-state automaton which receives sensorial and central process information. A task of garbage collector is used in a scenario with obstacles, using 3 robots, each one equipped with 14 infrared sensors and a communication mechanism with the central process. The available behaviors in the robots are: search-garbage, recognize-detected-object (obstacle, garbage object or robot), avoid-obstacle and put-garbage-object. In the simplest architecture, each robot has the same logic of behaviors but does not know the existence of the others. The cooperative behavior is emergent. The central process is executed in order to carry each garbage found at the collecting site. In the second architecture, each robot has a different

logic of behaviors and keeps the function of the central process. In the third architecture, the robot logic with the best results with regard to the previous architecture is reused in each robot. The robots report their location to the central process and ask for the location of the other robots. This solution improves the task performance and minimizes the inter-robot interferences. In the last architecture, the logic of the robot is maintained with regard to the previous architecture. The central process uses a regular grid of the scenario map for registering the visit frequency of each cell. If there is not new found garbage during a time period, the central process guides the robots to the cells with minor visit frequency (i.e. with a high probability of finding garbage). Each of the architectures was tested in simulation for knowing its performance and obtaining a comparison between the architectures. However, the architectures do not provide an explicit cooperation mechanism.
As a summary of precedent works, the Table 1 shows a comparative analysis between some important concepts related to the multi-robot systems. The table includes the MRCC approach presented in this chapter.

	Layered Archit.	Coordina-tion Mechanism	Inter-robot Communic.	Type of System	Role Concept	Formation Concept	Behavior-Based	Negotiation Mechanism
(Asama et al., 1989)			✓	Distributed				
(Mataric, 1995)			✓	Distributed			✓	
(Parker, 1998)			✓	Distributed			✓	
(Simmons et al., 2001)	✓	✓	✓	Distributed / Centralized	✓	✓		✓
(Weigel et al., 2002)		✓		Centralized	✓	✓	✓	✓
(Lima & Custódio, 2005)	✓	✓	✓	Centralized	✓	✓	✓	
(Sgorbissa, 2006)		✓	✓	Distributed	✓	✓		
(Quiñonez et al., 2009)				Distributed			✓	
(De la Rosa & Jimenez, 2009)				Distributed / Centralized			✓	
MRCC (Gonzalez et al., 2010)	✓	✓	✓	Distributed	✓	✓		✓

Table 1. Comparative analysis between related precedent works.

3. MRCC conceptual model

This section is focused on the application of the Multi-Resolution paradigm in the context of cooperative agents and robots. The conceptual model of Multi-Resolution Cooperation Control (MRCC) proposes that the cooperation control architecture be formed by a

hierarchy of layers, each one dealing with goals of different level of abstraction. For the case of robotic soccer, the MRCC composed by 4 layers appears as a complete solution; in this section this specific model is explained; so far in the chapter, it will be shown how it is used in the context of cooperative soccer robots.

3.1 Resolution paradigm

One of the most interesting multi-resolution models was proposed by Meystel & Bathija in the context of path planning for mobile robots (Meystel & Bathija, 2002). In order to get to a goal point, a robot must evaluate a great number of possibilities when trying to choose the appropriate atomic movement action to execute; at this decision level the resulting search tree can be enormous. Besides, because of the information volume, the set of data obtained from the environment and the incertitude involved in its interpretation, a long term plan is almost impossible to follow without adjusting it; each a modification of the plan is required, a new complex planning procedure should be executed. Due these reasons, Meystel & Bathija proposed a hierarchical model that decomposed the problem space in different resolution layers in a recursive way. At higher levels a coarse grain is used, thus making the search tree smaller; the planning task becomes faster and produces a short path including the main milestones that the robot must traverse in order to get to the goal position. Once the coarse grain path has been determined, each step of this path is analyzed using a finer grain resolution; a new planning procedure is executed only in the area restricted by the precedent planning resolution level. In consequence, the low level search task is reduced; only the area defined by higher levels must be considered. In the lower level, the planning task is applied to the atomic movement action decision; however this task only considers the areas restricted by all the higher levels. Each layer acts as a parametric and limiting body for the next lower layer, in such a way, the problem passing each of the layers decreases its complexity. In conclusion, thanks to the use of resolution models a more efficient path planning execution is obtained. Moreover, the planning at low levels can wait until it is really required; thus, re-planning can be done at the adequate resolution level and in the right moment.

3.2 Resolution applied to agent cooperation

The MRCC aims to apply the paradigm of decomposing in resolution levels the control a set of robots performing a complex task. From a structural point of view, this problem has the same nature as path planning. At each moment, the control system selects the next action that each one of the robots must execute in order to achieve the goals of the team. At the low level, the search tree is too big; there are too many possible alternatives, not only for the variety of available actions, but also for the possible orders of concurrent execution of these actions.

MRCC use the multi-resolution paradigm by performing a goal based decomposition of the control system into layers. The higher layers are responsible for the general team's goals, while lower ones take into account more specific goals that usually involve a reduced number of agents. In the general model, the highest layer takes decisions analyzing the system as a whole; the lowest layer control each individual agent in order to comply with the goals and responsibilities of a specific role. The intermediate layers, restricted by higher ones, determine the roles that should be assigned to agents. Figure 1 illustrates this model.

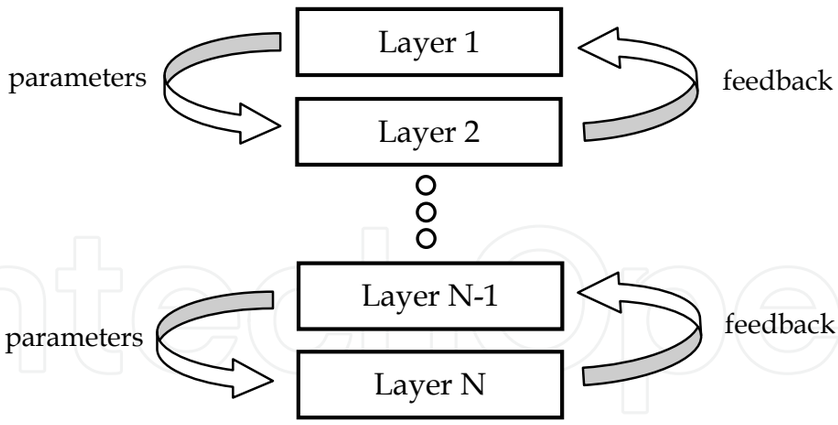


Fig. 1. General MRCC Model.

In other words, the concept of multi-resolution implies that a problem can be decomposed in different resolution layers without any restriction over the exact number of layers. In fact, the problem must be decomposed into as many layers as considered necessary, until reaching the desired atomic action decision level. MRCC being a result of the integration of cooperation control and multi-resolution gives the possibility to decompose the system into so many layers as wanted.

In general, there can be as many layers as desired. In practice, the system should have at least 3 layers; at least one intermediate must be present working as a bridge between the general goals and the individual ones. The relations inside the model refer to the connections and influences that exist between layers. The definition of parametric relation appears when a higher layer determines the guidelines that influences and restricts the way a lower layer takes decisions, so only a higher layer can do a parametric action over a lower layer. In the other side, the definition of feedback relation allows a lower layer to return information about the results obtained; this feedback can be used for the higher layer to make adjustments to its way of working. In a simple approach this relations can be established only by consecutive layers; however, in a more general approach, the parametric influences can be obtained from any of the higher layers, and the feedback information from any of the lower layers.

3.3 MRCC 4 layers

The first 3-layers MRCC proposed model includes the system, micro-social and agent layers. The intermediate micro-social layer aims to control the execution of cooperative actions between a reduced number of robots; at the agent level, robots assume cooperative roles assigned by the micro-social layer. This approach is suitable for robot teams where structuring the deployment of team members in the work space is not mandatory.

Afterward, a 4-layers MRCC model has been designed (Fig. 2), which tries to resolve the problems that were found in the precedent model. The first problem is the lack of overall structure of the MAS; the second one is that the agents don't get the robots to meet the required preconditions to start a cooperative action as often as desired. The function of the additional formation layer is to manage in a dynamic way structural roles associated to spatial regions. In order to solve the first problem, the control of team structure is incorporated into the system by raising negotiations between software agents, which represent spatial regions. Each region will control a reduced set of robots by assigning

structural roles to them. If the goals linked to these roles are designated to make the robots achieve the preconditions of the cooperative actions, the second problem would be also solved.

Therefore, by including a formation layer between the micro-social and the system layers improves the detection of cooperation opportunities, and allows having a more detailed decomposition of the goals of the system. In this way, the resources are used in a more efficient way and the achievement of the team goals is assured to be more suitable. Moreover, the inclusion of this layer in the system, allows the use of the model in other application contexts. The control of more complex cooperative systems is now possible.

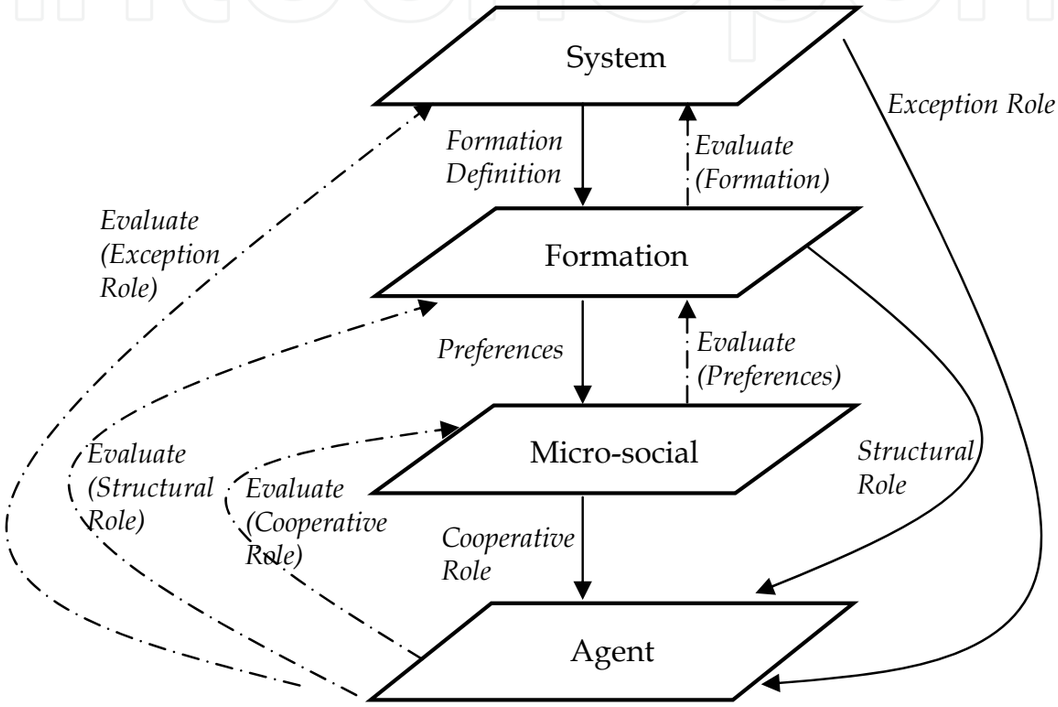


Fig. 2. The 4-layers MRCC Model.

A more precise description of each one of the layers in figure 2 is as follows:

System Layer: it is responsible for the decisions of the highest degree of abstraction (Gonzalez et al., 2007), where the global team goals are considered. It sends parametric influences to the lower layers; it is responsible for the general composition and the structure of the spatial regions of the formation level; it can also give more preference to some cooperative actions depending on the state of the team in relation to its goals. This layer can be seen as the coach of the team that takes strategic decisions.

Formation Layer: it manages a set of spatial regions in which actions take place; for each region a software agent aims to achieve a set of local goals. Robots can be assigned dynamically by a negotiation mechanism between regions as the goals of a region are not fulfilled. Inside a region, robots assume structural roles, which determine restrictions in the movement of a robot and also individual goals to accomplish. A structural role can be seen as a default role, assigned only when there are no opportunities to participate in a cooperative action.

Micro-social Layer: it is charged of the detection of opportunities to execute cooperative actions. Each robot includes a component that constantly monitors if the preconditions of any of the available cooperative actions are true. As an opportunity is detected, a negotiation protocol takes place between the concerned robots. If the negotiation succeeds, the action is executed by assigning cooperative roles to them.
Agent Layer: At this layer all the decisions taken at the formation and micro-social layers are transformed in actions performed by each individual agent according to the context and role assigned. Thus, each cooperative action of micro-social layer is carried out according to a number of roles that are assumed by an actor in a space of time. The appropriate role for an agent not only depends on the possibilities, the location and role it has assumed within the system (structuring), but also depends on the characteristics and abilities of the robot. If an agent is not involved in the execution of a cooperative action, its actions are guided by the structural role assigned by the formation level.
The concept of role is essential in the context of the MRCC. As explained by (Ch'ng & Padgham, 1998), a role can be defined as the set of responsibilities related to an agent, towards the objectives of the system. Moreover, according to what is mentioned in (Gonzalez et al., 2007), a role is a set of goals, skills and resources that enable an agent to perform specific tasks within the framework of collaborative action. Then, a role can be defined within the model as a set of elements (type of agent, responsibilities, skills, context and resources), assigned to an agent which allow to meet one or more individual goals aimed at supporting the objectives of the system. In the 4-layers model, there are 2 different kind of roles within the system, each associated with a specific aspect of the model, among them are:
Structural Role: According to Kendal defined in (Kendall, 1998) and (Kendall, 2000), and over the formation level, a structuring role is a set of defined characteristics to meet specific needs within a system. Among these needs are also found the interaction and fulfillment of responsibilities in the case of the formations level, compliance responsibilities in a spatial region.
Cooperative Role: The cooperative role is defined as the set of guidelines to be assigned to the agent to accomplish with part of the goals involved in the achievement of a cooperative action.

4. Dynamics of micro-societies

For the resolution and fulfillment of goals, it is possible to perform different actions on an individual basis, but depending on the context and situation, there may be events which need to be answered by one or more agents, in these situations the concept of cooperation is evident. In the MRCC model, the central mechanism to take advantage of these opportunities is the cooperative action execution control. In this section, a more detailed presentation of the procedure used in the micro-social level to manage cooperative actions is presented.

4.1 Cooperative actions

Cooperation in the context of multi-robot systems, and specifically when the multi-agent approach is used, refers to the interaction between players, performance improvement by making efficient use of team agent skills, and available resources (Weiss, 1999). This concept arises from the need to make a better use of the agents in a MAS and its features for both

work together to meet their individual goals. In situations where cooperation occurs, each of the agents involved must perform a set of actions to be completed within a certain time. In other words, the actions must be conducted concurrently in order to reach the goals (Gonzalez et al., 2007). The construction of a cooperative action model is quite complex, and can become a really difficult scenario when put in a changing and dynamic environment; at each moment, the general context can change and the system as whole must react and make decisions. In such scenario, agents must perform actions, cooperate and share resources in a very dynamic and opportunistic fashion, what makes the problem too complex to be solved in an understandable, simple and effective way.

In the MRCC approach, the proposed solution is to decompose the problem in different perspectives or levels of resolution. In this way, the overall goals of the primary objective is handled at higher levels of abstraction, and a more specific level will be solved more detailed aspects of the problem. As seen in the precedent section, the micro-social layer plays an essential role to connect the decisions of the system level with the individual agent. In this connection, the central component is the cooperative action execution mechanism.

4.1.1 Cooperative action's components

A cooperative action aims to fulfill a particular goal by a reduced group of agents by detecting when a cooperation opportunity is present. Once an opportunity is detected, a cooperative strategy is applied, which involves specialized negotiation mechanisms and also action preemption control. In general, a cooperative action includes the following components:

Opportunity Detection: this mechanism is responsible for assessing whether there are conditions necessary for a cooperative action, either taking into account all possible agents involved or only those who are considered as actors of the cooperative action.

Synchronization protocol: This mechanism allows control of the negotiations that occur before the beginning of the execution of the cooperative action. It can be done in one direction or two directions, by, sending the request and response or simply by sending request and confidence in the acceptance and timing of the agents. It is possible to use more complex protocols as the 2PC (two phase commit protocol) proposed in the context of concurrent transactions.

Expropriation: This mechanism evaluates whether the actual action that an agent is performed should be preempted by another more promising cooperative action. There can be different ways of measure how promising an action is, and also different algorithms to make the expropriation decision.

Monitoring: It is the mechanism necessary for every agent to follow the evolution of action in relation with the changes in the environment, in order to detect the moment of completion of the action, whatever the outcome. The action must end properly in case of failure, but also when it succeeds.

Rating: it is a qualifier mechanism by which measures the performance of the implementation of the cooperative action. Its use is usually related to learning tasks and providing feedback to higher levels.

4.1.2 Phases of a cooperative action

A cooperative action requires an interaction that takes place between the involved parties to define the roles to be assumed for each on during its execution. Consequently, a cooperative action required to comply with certain steps that are presented below:

1. Opportunity detection to execute a cooperative action, comparing the current situation (context) with a state of ideal conditions required; in other words, a measure of the matching between the action preconditions and the actual environment is calculated. If the matching is high enough, the layer will try to establish a micro-society
2. Creation of a micro-society, through a negotiation protocol where agents are invited candidates to participate. The agent that has detected the opportunity send an invitation to the other agents that are candidates to have the roles associated to the cooperative action. While analyzing the invitation, action preemption can occur.
3. Once the micro-society has been created, the execution of the cooperative action is accomplished as each agent carries out its goals according to the role assigned in the context of the micro-society.
4. When an agent terminates its participation in a cooperative action, it must exit. Not all the agents should remain in the micro-society until the end of the action; if an agent has fulfilled the goals associated to his role, it is free to do other actions.
5. Completion of cooperative action as a whole when the action goal is met or when it is not possible to complete it. Thus, agents has to monitor the evolution of the action execution in order to detect any of possible end conditions. Once an end condition is reached, an end action protocol is performed between the agents that still participate in the micro-society.
6. Assessment results of the execution of the cooperative action are calculated that can be used for learning or feedback purposes.

Notice that these steps are performed in a distributed, concurrent and dynamic way, there is not a unique coordinator of the execution of an action. In fact, all agents must include components that work permanently to detect opportunities, reply to invitations, assume assigned cooperative roles and monitor action execution. This distributed approach is very well suited for multi-robot applications, where each cooperative agent is embodied in a robot.
These series of steps are carried out to define the beginning and end of a cooperative action. The way they are carried out cooperative actions is generally regulated by a higher level, which gives preference parameters and values to respond to the detected opportunities. In particular, these preferences are taken into account while applying the action preemption strategy.

4.2 Cooperative strategies

There are different negotiation protocols and preemption politics that can be used while following the different steps involved in the execution of a cooperative action. Therefore, there can be a great variety of mechanisms to create and end micro societies; a particular instance of such mechanism is called a "cooperation strategy". The team situation in relation to its goals and the environment are characteristics that should be taken into consideration when choosing or designing cooperation strategies. In brief, a cooperative strategy incorporates the management of the mechanisms required in the execution of a cooperative action. In order to have a clear conception of what a cooperative strategy is, three different ones are introduced in this section.

4.2.1 BCA – Bind-based Cooperative Actions

This strategy is inspired from the human relationships that lead to work as a team. The idea is that the partnerships between agents are built through the analysis of the previous results

obtained when trying to carry out a cooperative action. An action can end either by a failure or by success, and then strengthening or weakening the cooperation binds between the involved agents.

In the BCA strategy (Perez, 2008), the negotiation protocol is very simple, only the invitation message is sent from the agent that has detected the opportunity to the other candidates to participate. If the communication channel is not reliable, a reply message can be included in the protocol; the purpose of this second message is only to acknowledge the reception of the invitation, but not to inform if the invited agent would participate in the micro-society. The action preemption politic is based on an elitist approach. The "probability of success" of all possible actions is calculated, and then the action with the higher probability is selected. In order to make this calculation, not only the information available in the invited agent is taken into account, but also the information sent in the invitation. In this way, the preemption mechanism uses the perspective of both agents in relation with the candidate action.

This simple protocol is based in an optimistic approach; the agent that sends the invitation hopefully waits that the other agents will get into the cooperative action. The fact receiving an invitation makes an agent aware of an opportunity, provides information about the possible action and leads him to participate in the micro-society; however the agent is not forced to get in the action. A possible scenario occurs when the initiator agent starts acting applying its role, but other agents don't. In this case, the evolution of the action can derive in two alternatives. In the first, the individual actions of performed by the initiator agent lead to a situation that seen from the other agents make them obtain a better value for evaluation of the action, which finally makes them participate. The other alternative is that the initiator agent perceives that the calculated probability of success goes down, leading to the preemption of the action. In both cases, the system continues without any problem.

4.2.2 CCS - Commit Cooperation Strategy

The BCA strategy is fast and simple, but has two problems related to the commitment. First, the agents do not have commitment with the micro-societies; thus, agents can start to execute actions that don't contribute to real cooperative action, if not all the required agents participate the goals are not fulfilled. In the other hand, there is a lack of persistency in the execution of an action; if there are two or more actions that have a similar score in the probability of success, the agent can easily switch between them. It would be better if the agent try to stay longer doing the same action, having more commitment with the action that is currently executed.

The CCS strategy (Gonzalez et al., 2006) aims to solve these two problems. The protocol used is a 2PC (two-phase commit), as the one that is usually used in distributed transaction systems. This protocol produces a dialog that includes four messages between the initiator agent and each one of the invited agents. If the 2PC protocol succeeds, all agents have confirmed their participation, and also all know that the others have agreed. Thus, it is assured that all are going to assume the proposed role and start acting accordingly. Additionally, the action preemption politic gives more importance to the cooperative action that is currently executed. A threshold based mechanism is used to implement this politic.

4.2.3 CCNet - Cooperation Contract Net

The CCS strategy (Pachon & Ariza, 2007) solved the problems detected in the BCA one. However, the requirements to establish a micro-society are too high. Thus, it becomes

harder to start cooperative actions; before starting, everything has to be almost perfect. As a result, when an action starts will probably lead to a success. Nevertheless, the problem is too few actions are initiated.

The CCNet strategy aims to obtain a balance between the former alternatives. In the negotiation step, the traditional and well known "contract net" protocol is used. As an agent receives an invitation, it evaluates a profit/cost function. It measures the profit obtained if the proposed action is selected and compares it against the cost of the preemption of the current action. All the invited agents send their values, and then the initiator agent makes a global evaluation. If this evaluation is good enough, the agent inform the others that the action will be executed; finally, the agents perform the as assigned role.

5. Validation model

The MRCC model is being validated through a practical implementation allowing setting up different experimental environments and protocols. In this section, the considerations to take into account in order to implement cooperative actions under the MRCC model are presented. Additionally, an introduction about how this model is being tested in the robot soccer domain is also included.

5.1 Elements of MRCC cooperative actions

In the precedent section the main concepts related to the execution of a cooperative action were introduced. The general internal architecture that is used to implement MRCC based agents was described in a precedent paper (Gonzalez et al., 2007). In this section, the key elements of this architecture that allow implementing in practice a cooperative action are explained.

Matching: It measures the similarity between the ideal and the current situations. A situation is defined from factors that should be considered in the cooperative action, such as object/agent positions, lengths, angles, etc.

Mapping: it makes all the decisions and calculations concerning the necessary actions that should be accomplished in order to reach the goal associated to a specific role.

Parameters: These are a set of input values that allow to specify the characteristics of matching and mapping functions. Thus, these parameters allow reusing the code of a cooperative action to achieve different goals.

Role: In practice a role is composed of mapping and matching functions and their parameters. The conjunction of these elements is used by the opportunistic detection mechanism and the decision component associated to a role (cooperative or structural).

Cooperative action: It aims to define the sets of roles that must participate to accomplish an action involved several agents; the role's matching is used to calculate if the cooperative action is suitable for the current situation or not.

Based on the above definitions, the process to create of a new cooperative action includes the following steps:

- To define the system's goal, so all the cooperative actions that will be defined in the system has to contribute to reach this goal.
- To define the action's objective which has to be aligned with the system's goal.
- To define all the roles needed in the action, as well as their matching and mapping functions.

- To identify how to calculate the cooperative action's matching, using the matching of each of its associated roles.
- To try, if possible, to reuse precedent roles, matching and mapping functions that already exist.

5.2 Study case: robotic soccer

Soccer is one of the most popular sports around the world; currently the FIFA has 208 members around the world (FIFA, 2010). Due to this popularity, intrinsic cooperation and dynamism of this sport did that different researchers get together and they created events where robotic soccer tournaments take place: Federation of International Robot-soccer Association (FIRA, 2010) and Robot World Cup Initiative (RoboCup, 2010). Each one of these events has its own rules and categories.

Despite the difference among categories, a team wishing to participate in any category, not only has to deal with the physical constrains of the category, but also with the inherent soccer problems as:

Limited communication: The existent communication systems are not reliable, due to some information could be lost when the transmission is occurring. Also the bandwidth of the channel is constraint by the technology, making that a message has to be cut in packets.

Best action vs. Time to find it: Often find the best action make necessary analyze all the possible actions and their consequence. However, this analysis requires time, but the soccer dynamism makes impossible to spend a lot of time to make a decision; for example, an action could be feasible in a certain moment but not later.

Limited resources: Two of the most important resources in soccer are the ball and the physical space. Robots occupy physical space, making sometimes impossible that a robot could receive the ball in an estimated position; especially when there are antagonist robots.

Cooperation: The speed that can reach the ball is always greater than the speed that any player could reach during the match. So, the team that takes advantage of this is the team that has most opportunity to win. This characteristic makes necessary to organize and coordinate the players.

5.3 Robot soccer examples

The success or failure of a robotic soccer team depends on the effectiveness and the appropriate number of cooperative actions that the system could perform according to the current match situation. The general goal of a robot soccer team is to win the game, which can be decomposed into two second level goals: score goals and avoid opponent goals. Therefore, it is important to create cooperative actions for both attacking and defending situations.

5.3.1 Building block

The goal of this cooperative action is to organize the robots in order to improve the team's defense. This is the base of other cooperative actions, because just changing the mapping and/or matching parameters, it is possible to obtain a variety of cooperative actions. In this action, the *neighborhood* defines the area where the agents will be taken into account to build the block; and the *figure* defines the geometric figure that the agents will be tried to form. This *neighborhood* is defined as a geometric figure, such as circle, rectangle, etc. The following concepts are used in this action:

Cardinality: the number of the agents that will be considered for the figure. If the number of the agents inside the neighborhood is less of this number, this action is not considered by the system. On the other hand, if the number is more than the cardinal number, this action will be penalized.
Triangular block is an example of this action, with the following characteristics(Fig. 3):
Neighborhood: the area where candidate agents are detected is rectangular.
Figure: the shape of the block formed by the agents is triangular.
Cardinality: requires 3 agents.
Number of roles: 3 instances of only one role is used, it knows how to go to the assigned point of the figure.
Mapping parameter: only one parameter is needed representing the ideal designed position for the roles, which can dynamically be changed by the system.
Role mapping: It aims to reach the point defined in the mapping parameter.
Role matching: The ideal situation is that each agent is over the desired position but not all the times happen this. Thus the Gaussian function is used to calculate the probability that an agent go to the desire position. The input for this function is the current position of the agent and the expected value of the Gaussian function is the ideal position where the agent is supposed to go, the variance is established by the programmer.
Matching parameters: These parameters are agent's positions P1, P2 and P3, the neighborhood area, the cardinality, the expected value and the variance.
Action matching: Initially, an agent calculates the probability to be in the desired position P1 (figure 3); if the probability is higher than a threshold then the agent gets all the teammates being in the neighborhood and estimates which agent is the best for each position. Once the best agents are established, then all probabilities are averaged to consolidate one action success probability.

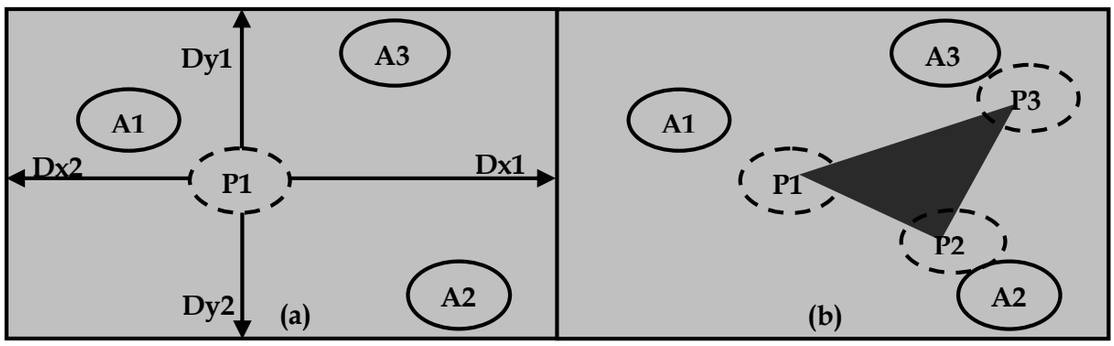


Fig. 3. Triangular block action. The point P1 is given in global coordinates, while points P2, P3, and the neighborhood are given as vectors from P1. A1, A2 and A3 are agents. a) The rectangular neighborhood definition, distances have not been equals. b) The points of the block which do no have to be in the neighborhood.

In the figure 3, it could be seen the concepts and definitions used for this action. Figure 4 shows the simulated results obtained for this action. It could be possible to change the role's mapping to obtain a block that follows the ball position, as in figure 5, where the idea is to take away from the ball but trying to maintain the triangular formation.

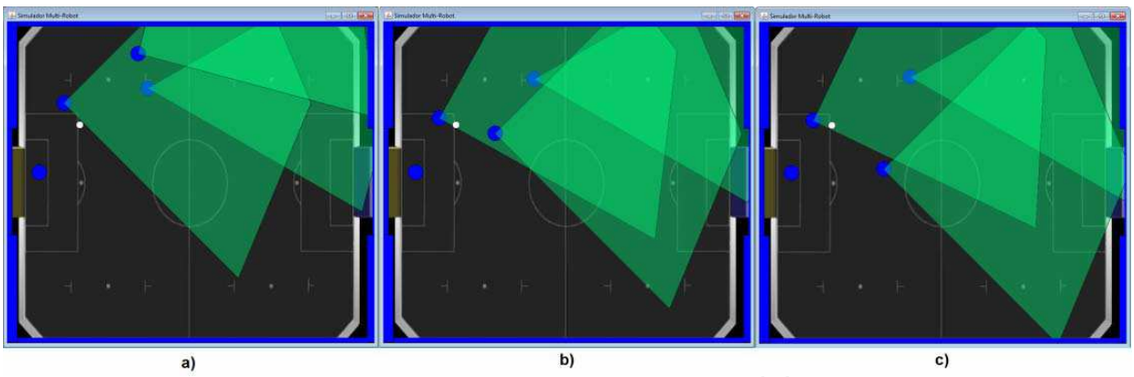


Fig. 4. Triangular block example. The blue spheres represent the robots, and the green area represents their vision area. a) The initial position of robots, b) how robots try to go to their position, and c) the final position.

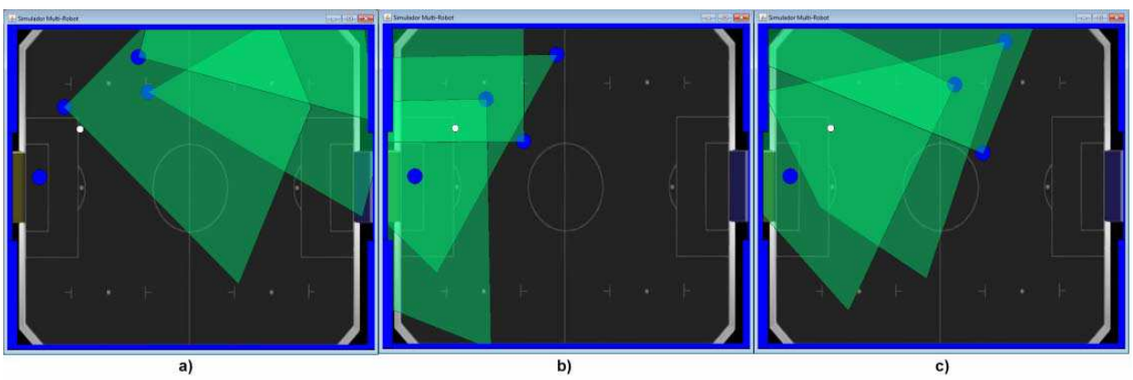


Fig. 5. Triangular block variation. The robots try to get as much distance as they can from the ball but trying to maintain the triangle. a) The initial position, b) forming the triangle, and c) the agents maintain the figure but they are taking away from the ball.

5.3.2 Pass

This action is one of the main plays to attack the opponent. It takes advantage of the velocity that the ball can reach when is kicked out by a player. The goal of this action is to send the ball to a teammate which is better positioned in order to score. This action has the following characteristics:

Number of roles: two roles are needed: passer and receiver.

Passer mapping parameter: It is the point where the player has to shoot the ball.

Passer role mapping: It calculates the point where the player has to kick the ball from the desired position.

Passer role matching: The ideal situation is that the agent is the closest teammate to the ball. If it is not the closest, it is not considered as a passer. The time needed for the closest opponent to the ball decreases the probability of the agent to be a passer. Additionally, this takes into account the angle that the agent has respect the opponent's goal (Fig. 6).

Receiver mapping parameter: It is the point where the player has to receive the ball.

Receiver role mapping: It tries to go to the point defined in the mapping parameter.

Receiver role matching: The ideal situation that all the opponents are not near to the ball trajectory, the receiver agent is close to the opponent's goal and the ball's trajectory is not

too large. Also, this function calculates the point where the agent has to receive the ball, as shown in figure 7.

Matching parameters: It is the distance from the receiver to the point and the parameters for the functions used in all matching.

Action matching: Initially, the agent calculates the probability to be a passer, if the probability is higher than a threshold then the agent get all the teammates and try to find which the best agent to receive the pass. Once the best agent is established, then all probabilities are averaged to consolidate one probability of success.

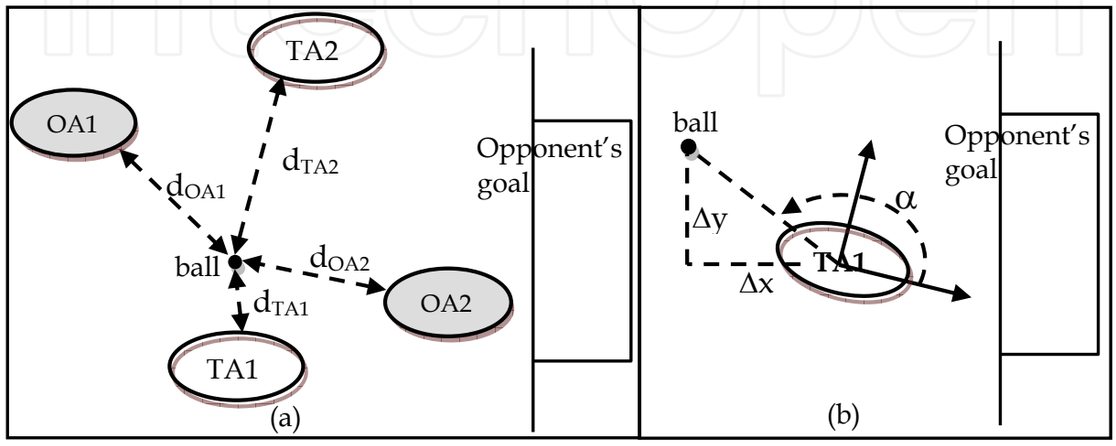


Fig. 6. Variables considered for passer role matching. a) Distance of all the agents to the ball. TA agents are teammates and OA agents are opponents. b) Orientation of the ball respect to the agent. If the agent is between the opponent's goal and the ball, the probability decreases.

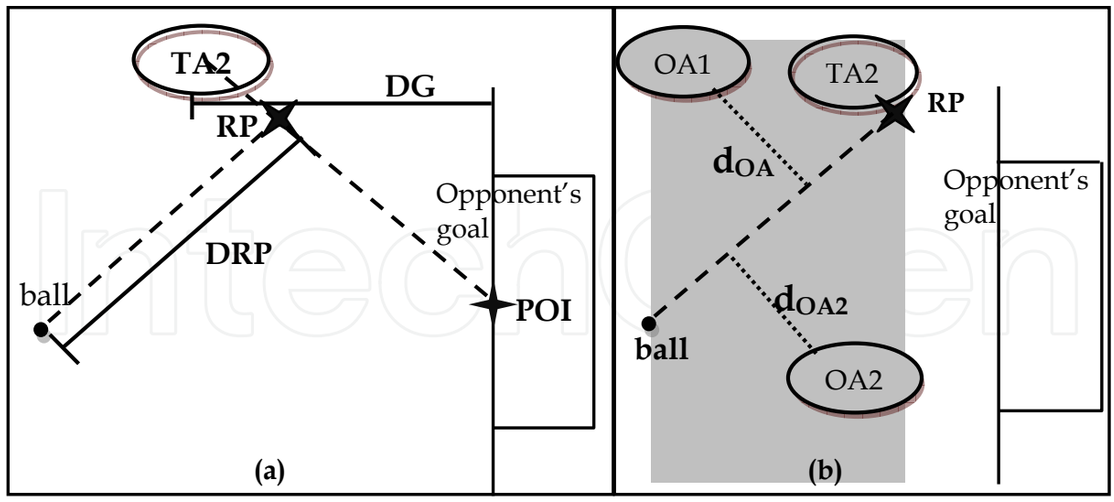


Fig. 7. Variables considered for a receiver agent. a) Calculation of the receiver point (RP) from the receiver's position to shoot the ball in direction of the point of interest (POI). DRP is the distance from the ball to RP and DG is the distance from the agent to the opponent's goal line. The probability to be a receiver is inversely proportional to the longer distance DG and DRP. b) Region to evaluate opponents near the ball's trajectory and their respective distance.

6. Simulation results

Below are shown two cases where MRCC was used as a cooperation control model. Over this model was implemented Commit bases Cooperation Strategy (CCS) (Gonzalez et al., 2006), Cooperative Contract NET (CCNET) (Pachon & Ariza, 2007), and Bind based Cooperative Actions (BCA) (Perez, 2008) as cooperation strategies. The two examples used to test MRCC were the Hunters–Prey problem and the robotic soccer.

6.1 Hunter-Prey problem

The hunters-Prey problem and the Robocup simulations were made using MRCC. A multirobot simulator has been built, where the agent control is implemented using the BESA agent framework (Gonzalez et al., 2003). The required mechanism to simulate the capture of preys has been added. The MRCC was used to manage the access to the physical world from the agents' brains, and also to facilitate the communication among them. Figure 8 shows a view of the simulator where agent systems are implemented and tested. Agents can be configured to have a limited field of view, which is represented by a triangle. Some simulation results were obtained after resolving this problem under different conditions (80 worlds of 12, 16, and 20 agents).

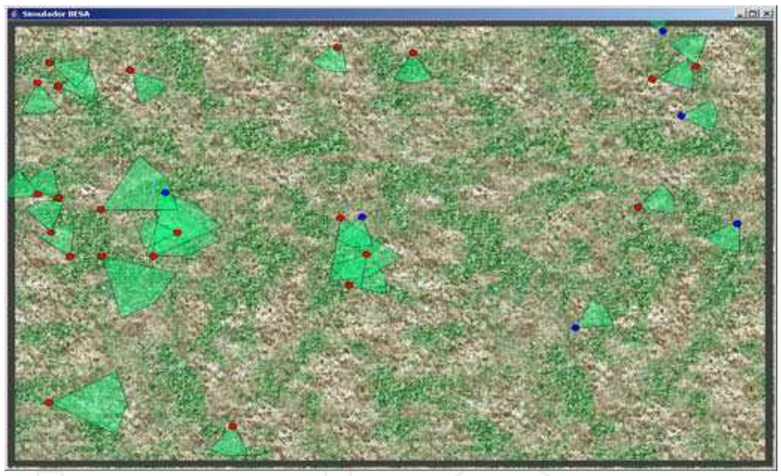


Fig. 8. Hunter-Prey simulation using MRCC.

The threshold to obtain the results was the time spent by the hunters to capture their prey. After analyzing the experimental results, it can be noticed that the number of built societies is greater when the number of hunters is superior to the number of the preys. In addition, when the number of agents increases, the number of micro-societies also increases proportionally. Also it was possible to observe that the success of the micro-societies is not related to the proportion between preys and hunters; but related to the number of conformed micro-societies, approaching the 50 percent of effectiveness.

The experiments have been run using the CCS approach (Gonzalez et al., 2006), and have been compared against an egoistic strategy, where hunters try to continuously persecute preys. The results show an increase in the performance when using the cooperative approach proposed by CCS. When analyzing the way micro-societies evolve, it was observed that the cooperative actions that involve more agents are very hard to achieve. In fact, the requirements imposed by the hard engagement of CSS are not easy to attain when

several agents are involved. However, it was observed that when a micro-society is established, there is a good probability of succeeding.

6.2 Robot soccer simulation

Pachon & Ariza show the results of the comparisons between CCS and Cooperative Contract NET (CCNET) as a strategy to create micro-societies using the MRCC model in order to organize robots that play soccer (Pachon & Ariza, 2007). Soccer simulations were made dividing a soccer field in sections and creating 4 different scenarios using CCS and CCNET (Fig 9).

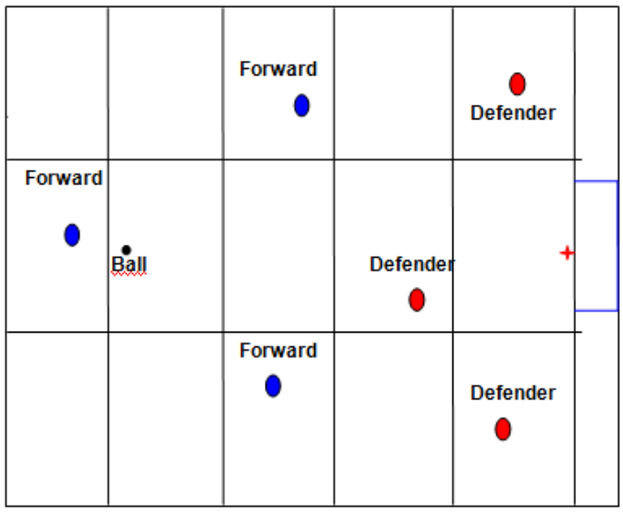


Fig. 9. Definition of micro-societies using CCS and CCNET applied to robotics soccer.

There was a greater attempt to set up micro-societies in CCNET. However, CCS was more effective in creating and implementing them. This allows the conclusion that CCS has higher rate to establish cooperation, but for robotic soccer CCNET is a better option to finish a cooperative action, which is a positive feature. Having a higher rate of dissolution of cooperative actions is desirable in a context of robot soccer, because the robots will have a faster response to change actions and/or roles in an environment where the scenarios change in a fast way.

About matching, both approaches, CCS and CCNET, had a similar response; the mean in both situations was similar. This can be explained because both approaches were calculating matching over geometrical calculi.

CCNet showed a higher goal rate due the adaptability this approach has to changes.

The cooperative action that was chosen most of the time by the agents was shooting to goal, due to the scenarios favored this move instead of passing the ball.

Due to some restrictions in the "view cone" of the agents, they had to choose to realize single-agent actions most of the time, because they were not able to see other team mates. And the actions that prevailed in this situation were defensive actions.

(Perez, 2008) uses the cooperation Strategy BCA (Bind Based Cooperative Actions) over MRCC to select cooperative actions. In this case the cooperative actions selected were 6 different types of soccer moves such as attacking, defending blocking, etc. The results of this approach consisted in proving how a group of agents can bind to and specific cooperative

action due a preference factor, which is decided by the cooperative agents. Also a matching function works in this strategy that initiates cooperative actions. These actions are started only if a group of agents and the environment fill up the requirements to start such cooperative actions. Finally, the approaches CCS and BCA were tested and approach with machine learning.

7. Final discussion

Nowadays, a general trend is to use robots for more complex tasks. However, as complexity increases it is not viable to get the task done using a single robot. Even if we were able to build such a complex robot, there are situations where the use of several robots is mandatory. In fact, it is not viable to incorporate all the abilities and resources required for a complex task into a single robot. A better approach is to distribute these capacities into several specialized robots which could accomplish the task by adding their skills in a synergic fashion. Besides, there are many situations where the use of several robots will allow getting the work done in less time or in a more efficient way. Multi robot systems incorporate these properties: specialization and redundancy, which result in qualitative and quantitative gains in comparison with single robot approaches. Thus, multi robot systems are required to achieve complex tasks. Nevertheless, team control implies new problems to solve; a control strategy is required to determine who does what, at what moment and which resources can be used. Cooperation is the key, as it incorporates mechanisms to carry out: task allocation, synchronization and planning, and conflict resolution.

The MRCC, Multi-Resolution Cooperation Control, proposes a general solution to the cooperation control by decomposing it into a set of layers with different level of abstraction. The key idea is that higher levels influence the behavior of lower ones. The control complexity is split into the layers, each one is concerned with goals according to its abstraction plane. In general, there can be as many levels as required; an outline of the 4-level approach has been introduced. At the agent level, the lower one, individual teammates assume roles; these roles are assigned by the decision mechanisms included in the higher levels. The system level, the higher one, allocates exception roles in order to deal with special situations that affect the whole team. The formation level, through an organization based on zones, gives structural roles to agents. The micro-society level decides which agent should assume a cooperative role. In a similar fashion, the way cooperative actions take place and the structure of the formations are affected by the higher levels. Finally, the model also includes feedback interactions from lower levels to higher ones; thanks to this information, dynamic control and learning can be achieved.

This chapter was focused in the micro-social level. The central issue at this level is to execute cooperative actions, which aim to accomplish a goal between a selected and reduced set of agents. The detection of cooperative actions is based on an opportunistic approach. As an opportunity is detected a negotiation protocol allows to form a micro-society and specific cooperative roles are assigned to agents. This proactive mechanism is performed in a distributed way; each agent is looking for cooperation opportunities and invites others to form micro-societies. There is a balance between the autonomy of the agent and its commitment with the team. Each agent works in an autonomous fashion, there is not hierarchy between robots; but when the agent is required to assume a role, it changes its own goals for those of the role it has to play. At the end, what are more important are the team goals.

Actually, the MRCC has been implemented as a general cooperation framework. The case studies of hunters and preys and robot soccer have been studied and implemented. In the former one, a 3 layer MRCC model has been enough to have a good performance. However, in the soccer task, it is mandatory to incorporate the formation level. Experimental simulation trials have demonstrated the viability of the approach. Actual work includes the refinement of the formation level and the validation of the MRCC with real robots.
In the near future, it is expected to have a MRCC based team of robotic soccer participating in international competitions. The approach will be tested not only in competitions, where the desire of winning often leads to simplified assumptions of the problem and centralized control approaches. The MRCC operates in a real distributed system and is able to deal with restrictions concerning the sensor information available and of the communication capacity.

8. Acknowledgements

This work is product of the projects Cooperative Agents ("Cooperación en Sistemas MultiAgentes Aplicada a Robótica Móvil" and "Robótica Cooperativa Basada en Agentes Heterogéneos Aplicada a Educación en Tecnología") financed by the government of Colombia through COLCIENCIAS with the participation of Pontificia Universidad Javeriana, Universidad de los Andes, Maloka and Universidad del Norte. The authors thank the students and colleagues that have contributed to the development and testing of the architecture MRCC.

9. References

Asama, H.; Matsumoto, A. & Ishida, Y. (1989). Design of an Autonomous and Distributed Robot System: Actress, *Proceedings of IEEE/RSJ International Workshop on Intelligent Robots and Systems (IROS)*, pp. 283 – 290, Tsukuba – Japan, September 1989, IEEE and Robotics Society of Japan (RSJ).

Ch'ng, S. & Padgham, L. (1998). From Roles to Teamwork: A Framework and Architecture. *Applied Artificial Intelligence Journal*, Vol. 12, No. 2 - 3, (1998), page numbers (211-231), ISSN 1087-6545.

De la Rosa, F. & Jimenez, M.E. (2009). Simulation of Multi-robot Architectures in Mobile Robotics, *Proceedings of IEEE Electronics, Robotics and Automotive Mechanics Conference (CERMA)*, pp. 199 – 203, Cuernavaca – México, September 2009, IEEE.

Ferber, J. (1999). MultiAgent Systems: an Introduction to Distributed Artificial Intelligence, Ed. Addison Wesley, ISBN 978-0201360486.

FIFA. (2010). History of Football – The Global Growth. Available at: http://www.fifa.com/classicfootball/history/game/historygame4.html, July 2010.

FIRA. (2010). Federation of International Robot-soccer Association. Available at: http://www.fira.net/, July 2010.

Gonzalez, E.; Avila, J. & Bustacara, C. (2003). BESA: Behavior-oriented, Event-Driven and Social-based Agent Framework. *Proceedings of Parallel and Distributed Processing Techniques and Applications (*PDPTA), pp 1033-1039, Las Vegas - USA, June 2003, CSREA Press.

Gonzalez, E.; Vazquez, A.; Plata, A.; Montañez, L.; Perez, A. & Bustacara, C. (2006). CCS: Commit based Cooperation Strategy for MultiRobot Systems, *Proceedings of*

International Symposium on Robotics and Automation (ISRA), pp. 193 - 198, San Miguel – México, August 2006.

Gonzalez, E.; Perez, A.; Cruz, J. & Bustacara, C. (2007). MRCC: A Multi-Resolution Cooperative Control Agent Architecture, *Proceedings of IEEE/WIC/ACM Intelligent Agent Technology (IAT)*, pp. 391 - 394, San Francisco – USA, November 2007, IEEE.

Hershberger, D.; Simmons, R.; Singh, S.; Ramos, J. & Smith, T. (2002). Coordination of Heterogeneous Robots for Large-Scale Assembly, In: *Robot Teams: From Diversity to Polymorphism,* Balch, T. & Parker, L.E., (Ed.), page numbers (369-380), A K Peters Ltd, ISBN 1-56881-155-1.

Kendall, E.A. (1998). Agent Roles and Aspects, In: *Lecture Notes in Computer Science - Workshop on Aspect Oriented Programming – ECOOP 1998*, Goos, G.; Hartmanis, J. & van Leeuwen, J., (Ed.), Vol. 1543, page numbers (431-432), Springer, ISBN 978-3-540-65460-5.

Kendall, E.A. (2000). Role Modeling for Agent System Analysis, Design, and Implementation. *IEEE Concurrency,* Vol. 8, No. 2, (April 2000), page numbers (34-41), ISSN 1092-3063.

Lima, P.U. & Custódio, L.M. (2005). Multi-Robot Systems, In: *Innovations in Robot Mobility and Control,* Patnaik, S.; Jain, L.C.; Tzafestas, S.G.; Resconi, G. & Konar, A., (Ed.), Vol. 8, page numbers (1-64), Springer, ISBN 978-3-540-26892-5.

Mataric, M. (1995). Issues and Approaches in the Design of Collective Autonomous Agents. *Robotics and Autonomous Systems,* Vol. 16, No. 2 – 4, (December 1995), page numbers (321-331), ISSN 0921-8890.

Meystel, A. & Bathija, A. (2002). Multiresolutional Planning: Using the Randomized Tessellation of the State Space, *Proceedings of International Symposium on Robotics and Automation (ISRA)*, Toluca – México, September 2002.

Pachon, A. & Ariza, L. (2007). Cooperation Techniques based on Contract NET CCNET, Pontificia Universidad Javeriana, Bogotá - Colombia.

Parker, L. E. (1998). ALLIANCE: An Architecture for Fault Tolerant Multirobot Cooperation. *IEEE Transactions on Robotics and Automation,* Vol. 14, No. 2, (April 1998) page numbers (220-240), ISSN 1042-296X.

Perez, A. (2008). Learning Techniques in Multi Agent Systems applied to cooperation strategies. Master Thesis. Pontificia Universidad Javeriana, Bogotá – Colombia.

Quiñonez Y.; de Lope, J. & Maravall, D. (2009). Cooperative and Competitive Behaviors in a Multi-robot System for Surveillance Tasks, In: *Lecture Notes in Computer Science - Computer Aided Systems Theory - EUROCAST 2009,* Moreno-Díaz, R.; Quesada-Arencibia, A. & Pichler, F., (Ed.), Vol. 5717, page numbers (437-444), Springer, ISBN 978-3-642-04771-8.

RoboCup. (2010). RoboCup World Championship and Conference. Available at: http://www.robocup.org/, July 2010.

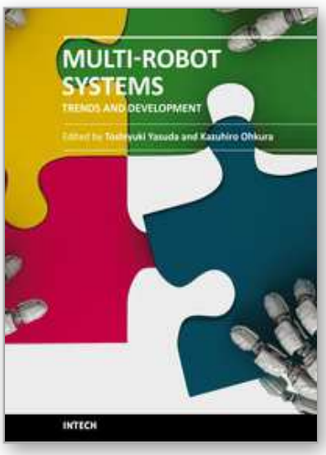
Sgorbissa, A. (2006). Multi-Robot Systems and Distributed Intelligence: The ETHNOS Approach to Heterogeneity, In: *Mobile Robotics, Moving Intelligence,* Buchli, J., (Ed.), page numbers (423-446), Pro Literatur Verlag, Germany / ARS, Austria, ISBN 3-86611-284-X.

Simmons, R.; Singh, S.; Hershberger, D.; Ramos, J. & Smith, T. (2001). First Results in the Coordination of Heterogeneous Robots for Large-Scale Assembly, In: *Lecture Notes*

in Control and Information Sciences, Thoma, M. & Morari, M., (Ed.), Vol. 271, page numbers (323-332), Springer, ISBN 978-3-540-42104-7.

Weigel, T.; Gutmann, J.-S.; Dietl, M.; Kleiner, A. & Nebel, B. (2002). CS Freiburg: coordinating robots for successful soccer playing. *IEEE Transactions on Robotics and Automation*, Vol. 18, No. 5, (October 2002), page numbers (685-699), ISSN 1042-296X.

Weiss, G. (1999). Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence, MIT Press, ISBN 978-0262232036.



Multi-Robot Systems, Trends and Development

Edited by Dr Toshiyuki Yasuda

ISBN 978-953-307-425-2
Hard cover, 586 pages
Publisher InTech
Published online 30, January, 2011
Published in print edition January, 2011

This book is a collection of 29 excellent works and comprised of three sections: task oriented approach, bio inspired approach, and modeling/design. In the first section, applications on formation, localization/mapping, and planning are introduced. The second section is on behavior-based approach by means of artificial intelligence techniques. The last section includes research articles on development of architectures and control systems.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Enrique Gonzalez, Fernando De la Rosa, Alvaro Sebastian Miranda, Julian Angel and Juan Sebastian Figueredo (2011). A Control Agent Architecture for Cooperative Robotic Tasks, Multi-Robot Systems, Trends and Development, Dr Toshiyuki Yasuda (Ed.), ISBN: 978-953-307-425-2, InTech, Available from: http://www.intechopen.com/books/multi-robot-systems-trends-and-development/a-control-agent-architecture-for-cooperative-robotic-tasks

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen