How a Cooperative Behavior can emerge from a Robot Team

A. D'Angelo[1], E. Menegatti[2], and E. Pagello[2]

[1] DIMI, via delle Scienze 206, University of Udine antonio@dimi.uniud.it
[2] DEI, via Gradenigo 6, University of Padua {emg,epv}@dei.unipd.it

In this paper, we suggest an hybrid architecture where the deliberative part takes advantages from the reactive one and vice versa, to make a multi-robot system to exhibit some assigned cooperative task. We explain our architecture in terms of schemas and a set of firing conditions. To experiment our approach, we have realized an implementation that tries to exploit the resources of our robot team participating to Middle-size RoboCup tournaments. Each individual exhibits both reactive and deliberative behaviors which are needed to perform cooperative tasks. To this aim we have designed each robot to become aware of distinguishing configuration patterns in the environment by evaluating descriptive conditions as macroparameters. They are implemented at reactive level, whereas the deliberative level is responsible of a dynamic role assignment among teammates on the basis of the knowledge about the best behavior the team could perform. This approach was successfully tessted during the Middle-size Challenge Competition held in Padua on last RobCup2003.

1 Introduction

The most recent robotics applications have shown a growing interest in developing colonies of robots within industrial and civil environments, switching robot design from the goals of controlled speed, high accuracy and repeatability toward new targets of flexibility and reliability. A key issue to successfully perform such kind of advanced tasks is figuring out how to make emerging cooperative abilities within this context.

The *differentiating* robot societies[15] show a large number of homogeneous individuals with limited abilities, whereas the *integrating* societies are usually characterized by a small number of heterogeneous and specialized individuals. Both societies include individuals with well-distinguishing skills referred to the role to play inside the group or the aptitude to modify dynamically its behavior while performing an assigned task.

A group of robots is a robotics team only if it exhibits the ability to perform cooperative tasks, providing better performance for their individual components and taking advantages from distributed sensing and acting. Soccer-robots International Games, like RoboCup [8], are very useful testbeds to experiment various approaches to coordinate multiagent systems operating in real environments. The solution, implemented at the IAS Laboratory of Padua University evolves from those successfully adopted in the ART Team on RoboCup-2000 [16], and now in the new Team Artisti Veneti [14] and [13].

2 Multi-robot Systems

A multi-robot system is characterized by attributes like its size, composition, and reconfigurability as well as its communication topology, availability, and range [11]. Also its collective intelligence [9] and agent redundancy are important features. Thus, a group of mobile robots gives rise to an intelligent multi-robot system if they cooperate to solve a given complex task by communicating among individuals and allowing dynamic group reconfigurability.

Robotics team design addresses issues such as the specification if each individual robot share or not a common goal [10] or the choice between distributed and centralized control. Nevertheless, communication among individuals cannot be ignored and, depending on *explicit* or *implicit* one, or a combination of both, the group exhibits very distinguishing behaviors.

In the next sections we shall insist between *explicit communication*, where signals are intentionally shared between two or more individuals, and *implicit communication*, by observing other robot actions. Despite what appears at a first glance, intelligent cooperation doesn't necessarily require an explicit communication among robots. For example, in our preceding papers [7], [12] and [14], we have exploited the case of forcing collective behaviors through implicit communication. There, the idea of perceptual patterns, recognizable by evaluating a set of scalar quantities, termed *macroparameters*, has been introduced. Every agent was equipped with a set of basic behaviors and, moreover, each behavior was defined with its *complementary* [6].

3 Behavior-based Approach

Developing robot agents includes both the design of physical components and the implementation of new software architectures with the aim of investigating the issues that arise from the integration of different software components which support the *decide-sense-adapt behavior cycle* and which, starting from the pioneeristic work of Brooks [5], is controlled by a set of behaviors.

This architecture, known as *behavior-based approach* but also termed *reactive control*, has become very popular along the time. It refers to the direct coupling of perception to action as specific technique which provides time-bound responses to robots moving in dynamic, unstructured and partially

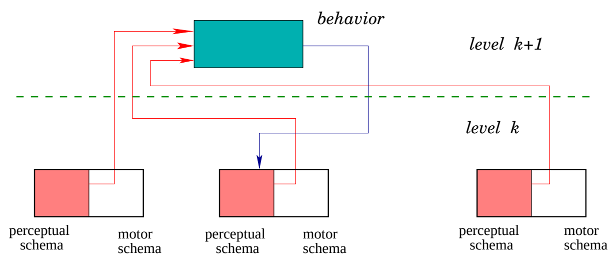

Fig. 1. Controlling the underline level

unknown environments. A behavior is defined to be a control law for *achieving* and/or *maintaining* a given goal. Usually, robot agents have multiple goals, including at least one achievement goal and one or more maintenance goals.

3.1 Schema Approach

The behavior-based approach assumes a robot to be *situated* within its environment. This means that a robot interacts with the world on its own, without any human intervention, namely, its perspective is different from observer's. Moreover, since robots are not merely information processing systems, its *embodiment* requires that both all acquired information and all delivered effectors command must be transmitted through their physical structure. Different research areas like biology, ethology and psychology, have contributed to the design of robot control. Among them, *schema*-based theories have been adapted by Arbib [2] to build the basic blocks of robot behaviors.

Originally, when they appeared during eighteenth century, they provided a mechanism of understanding sensory perception in the process of storing knowledge. Such a philosophical model to explain behaviors has also become an useful abstraction to implement behaviours taking advantage from the object-oriented programming. In this perspective, a *schema* is a generic template for doing some activity which is parameterized and created like a class (*schema instantiation*). Following Arbib [1] we implement a *behavior* with one **motor schema**, representing the physical activity, and one **perceptual schema** which includes sensing.

3.2 Implementing Schemas

Schema-based methodologies are largely used in robotics. So, *motor schemas*, as they were proposed and developed by Arkin [3], are the *basic units of behavior from which complex actions can be constructed; they consist of the knowledge of how to act or perceive as well as the computational process by which they are enacted* [4]. Each schema operates as a concurrent, asynchronous process initiating a behavioral intention by reacting to sensory information.

In Arbib and Arkin, all schemas are always active producing outputs as action vectors which are summed up, whereas our implementation assumes

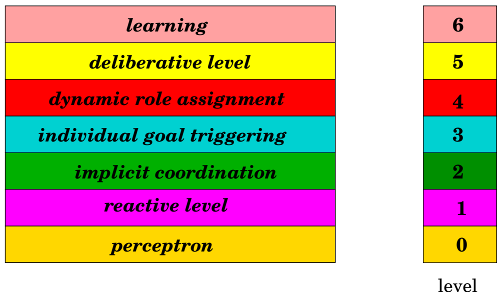

Fig. 2. Levels of Control

only one schema to be active at a time, in a winner-take-all fashion. Moreover, the output is not a continuous signal but either a motor command to feed some servo or an evaluated condition affecting the activation/inhibition mechanism for another schema.

In this perspective, the governor's unit of each robot is organized at many levels of abstraction, the lowest one being directly coupled with the environment by the robot servos. They are implemented as C routines which access sensor and effector devices of the robot. Each level is populated by a set of control units, which are schema-based *behaviors* receiving sensor information by monitoring the lower level and acting on some *releaser*. Fig. 1 makes explicit how conditional activations propagate through levels.

For example, simple behaviors like *defendArea* or *carryBall* are implemented in C as motor schemas accessing directly robot effectors. Now, we are able to build the two basic behaviors *playDefensive* and *chaseBall* by simply appending the two perceptual schemas as explained by the following behavior constructing rules:

$$\begin{array}{ll} playDefensive : & seeBall \rightarrow defendArea \\ chaseBall : & haveBall \rightarrow carryBall \end{array}$$

Also the perceptual schemas *seeBall* and *haveBall* are implemented in C by accessing virtual sensor devices like *senseBall* and *touchBall* which are fed by robot physical sensors. At any level, the primitive control component is a behavior with only a perceptual and motor schemas. By releasing a behavior we mean an *activation-inhibition* mechanism built on some given *evaluating-condition* basis. Thus, a primitive behavior results in appending just one perceptual schema to one motor schema so that, at reactive level, we have the *sensorimotor coordinations* the individual robot is equipped with.

The reactive level is the lowest control level because it uses only information coming from sensors and feed motors with the appropriate commands. Compound behaviors appear only at higher levels while they are receiving more abstract information about the environment as they are filtered by lower behavior functioning. As suggested by fig. 2, individual control has been organized into different layers, each of which represents a different level of abstraction such that an upper level risults in a more abstract handling of the environment. So, the second layer assumes that some perceptual patterns represent events generated by other individuals, either opponents or teammates.

Moreover, the corresponding schemas can control the underline reactive behaviors but, at the same time, they are also triggered by the individual goals every robot should pursue. The higher layers refer to the cooperation capabilities that any robot could exhibit as teamate while a cooperative behavior is going to emerge. We shall describe them in the next sections.

As a matter of implementation, we want to sketch some solutions we have devised to actually implement such an architecture. All schemas are executed as threads in the so called **ade** runtime environment, expecially designed for real-time systems over an Unix/Linux kernel. Also the arbitration module is executed as a thread; more exactly, three different threads have been committed to select a behavior for its execution on the winner-take-all basis.

Looking at fig. 3, it can easily understood how the governor's unit operates to control robot behaviors. First of all, sensor information, coming from different sources are piped towards the *sensor drivers* which work as input controllers. They provide all perceptual schemas with the required sensing, also feeding the C-implemented motor schemas which demand immediate sensor data for triggering. The modules labelled *brain*, *ruler* and *teamplay*, implemented as threads, are committed to select the most suitable motor schema to gain exclusive control of the robot. The thread *brain* evaluates all the possible activating conditions, implemented as **and-or** networks but organized to cover levels of abstraction. The *ruler* affects robot behaviors by adapting their execution to the constraints which stem from soccer play rules avoiding, as far as possible, violating situations.

At last, the module *teamplay* provides the necessary coordination a single teammate must exhibit to eventually make emerging a collective behavior inside the group, for example, a *triangulation* while passing the ball. We deal with this topic in the next section.

4 Building an Hybrid Architecture

As previously stated a schema is the building block of our architecture where perceptual components are organized into an hierarchy of abstraction levels. They feed motor schemas acting as either a control mechanism or a delivery device towards robot effectors, namely, the *wheel-driving motors* and the *kicker*. No effector is needed to control vision as it is implemented by a camera monitoring the environment with an omnidirectional mirror. At reactive level (cfr. fig. 2) schemas are true behaviors whereas at higher levels they work as triggering mechanism to modulate the whole behavior of any individual.

The actual implementation rearranges perceptual schemas in a network of **and-or** nodes, generated at startup by executing appropriate scripts de-

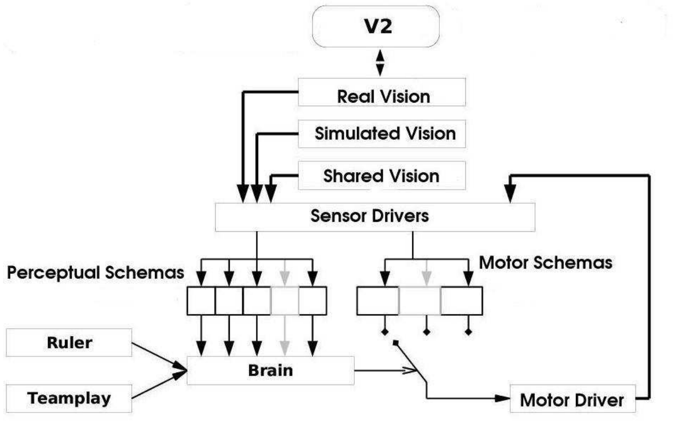

Fig. 3. General Architecture of a single Robot

scribing that hierarchy and which can be easily changed. So many different configurations have been tested during experimental trials in our Lab.

4.1 Integrating Deliberation

If we build robots only considering the reactive level depicted in fig. 2, we couldn't endow cooperation capabilities in our team because of the lack of any mechanism which allows a robot behavior to take into accounts the behavior of other robots. So, an *implicit coordination* is required to trigger individual robot behaviors in such a way some *actions that are a part of an agent's own goal-achieving behavior repertoire, but have effects in the world, help other agents to achieve their goals* [10].

However, a group of robots whose coordination is based on some *stigmergic*[3] property, may exhibit no collective behavior because stigmergy doesn't guarantee cooperation. To force an emergent collective behavior we could need to endow deliberation into the group of robots. Integrating deliberation within a behavior-based architecture is a current topic of research and, moreover, it is a matter of an active debate because the *reactive/deliberative* trade-off depends on how many representational issues are endowed and how much reasoning process is made available to the system.

Considering that any deliberative process slows down the *decide-sense-adapt behavior cycle*, a solution to this problem could suggest a different priority levels to be assigned to the different layers appearing in fig. 2 and which are mapped into different levels of abstraction. So, we have devised an hybrid architecture growing up from a level to the next level in such a way *the more is the number of levels the more are the deliberative capabilities of an individual*

[3]this term, commonly used in biological literature, refers to the animal capabilities to coordinate without explicit communication.

robot. At the top we have the learning process which, at the present, we have not yet implemented. Just a level below there is the effective *deliberative level* but this property becomes less feasible as we approach the reactive level.

There are also two intermediate levels which cope with the communication capabilities of the robot. The lower implements stigmergy whereas the higher deals with the dynamic role exchange, which is needed if we want an effective control on cooperation to be triggered by internal and external firing conditions. We could say that our robots are featured with both *reactive and deliberative communication*, the latter implying some form of negotiation.

In our preceding works we have tried to have the same result only using stigmergy, avoiding any form of negotiation. Macroparameters and quality functions have been the tools we have used to this aim. At the present we want better evaluating the two solutions.

4.2 Implementing Coordination

As previously stated, coordination has been implemented at two stages: the former, dealing with the reactive level, provides the necessary conditions to be verified to start an activation cycle of cooperations. Such conditions are evaluated acquiring information from the environment by testing for specified patterns. The latter is involved in coordination properly by examining and scheduling the behaviors which are the best candidates to cooperate with.

As an example, let us address the coordination between two robots with the task of carrying the ball towards the opponent goal, eventually passing and defending it from opponents' attacks. A number of conditions must be be continuously tested if we want such a cooperative task to become effective. First of all, at any stage of cooperation we should require the two robots to play well-specified roles. So, if we assign the *master* role to the robot chasing the ball, the latter can be considered the *supporter* of the former. Any role is played at different levels; let's call them *canbe*, *assume,acquire*, *grant*, *advocate* so that we can build the following coupled behaviors.

behavior **clampmaster**
$haveBall(\mathbf{me}) \& \neg haveBall(\mathbf{mate}) \rightarrow acquire(Master);$
$acquire(Master) \& Notify(Master) \rightarrow advocate(Master);$
$grant(Master) \rightarrow advocate(Master);$
behavior **clampsupporter**
$\neg acquire(Master) \& canBe(Supporter) \rightarrow assume(Supporter);$
$assume(Supporter) \& Notify(Supporter) \rightarrow acquire(Supporter);$

Here, because the role assignment depends on *ball possess*, we have used the condition *haveBall* to discriminate the robot which is really carrying the ball. It can be understood as a *macroparameter* [7], [6] in the style of our preceding works, that describes the characteristic of the environment and because it can be evaluated by different robots. Moreover, it resyncronizes the activation of a new cooperation pattern.

They are complementary behaviors and they must be arbitrated in such a way they must be assigned to the robots with the right referring roles. The basic rule is that a *master* role must be **advocated** whereas the *supporter* role should be **acquired**. To this aim, we require two reciprocity rules where a role is switched either from *acquire* to *advocate* or from *assume* to *acquire* provided that a **notification** is made to the referred teammate. Such rules imply a direct communication between teammates to negotiate the role on the *first notified/first advocated* basis as depicted below.

$$\begin{array}{l} notify(\mathbf{role}) \\ \quad Supporter(\mathbf{mate}) \rightarrow reply(\mathbf{role}, \mathbf{mate}); \\ \quad Master(\mathbf{mate}) \rightarrow request(\mathbf{role}, \mathbf{mate}); \end{array}$$

In such a way, the robot carrying the ball suggests a teammate to become supporter by advocating the master role and forcing the latter to acquire the supporter role. By so doing the former issues a behavior of *chaseBall* whereas the latter exhibits a behavior of *approachBall* and they work as a reinforcement to maintain or exchange these roles.

5 Experimental Results

Implicit coordination and role exchange are necessary tools for activating and tailoring cooperative behaviors. To tell the truth only implicit coordination is strictly necessary as it has been repeatedly pointed out in literature. The problem is how many times the interaction patterns are detected by different robots to initiate a cooperation task. In the case of simulated soccer games, we have shown [6] that a continuous evaluation of environmental patterns could fire ball exchange between teammates during attack. The number of succeded cooperations was made high by increasing the circumstances of positive activations by a kind of *brownian motion* among teamates. The situation becomes more difficult in the case of Middle-size robot competitions where the evolving dynamic of teammates cannot provide such a satisfactory number of active interactions. So, role assignment becomes a very important feature to be endowed into a soccer team. In a preceding implementation [14] we have forced cooperations by evolving *macroparameters* into *quality functions*. On the contrary, the current approach would exploit a different point of view by considering an active engagement of teammates during the phase of role assignment. As shown in fig. 4[4], for the last RoboCup International Competition, held in Padua on July 2003, we have developed some testing programs, that were effectively exhibited during the Middle-size Challenge tournament.

We had to show a cooperative behavior of two companion robots. As it can be easily understood looking at the figure, two soccer robots were involved

[4]at the location http://www.dei.unipd.it/~robocup/video/tenaglia.avi it is accessible the full video

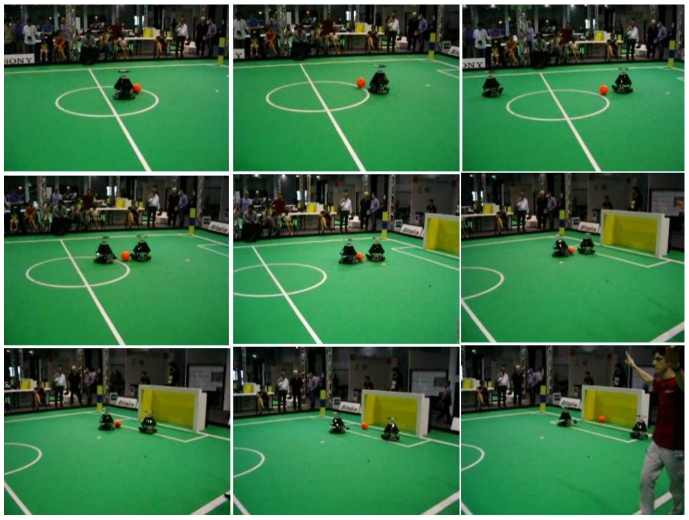

Fig. 4. Two attacking robots chasing the ball in a clamp

in a cooperative task which results in carrying the ball towards the opponent goal to score safely. The first robot is chasing the ball, whereas its companion is approaching to protect it. Its *approachingBall* behavior is a consequence of a sharp negotiation implemented by a low number of *short message exchanges*.

Thus, the two robots are moving in a strict coordinate behavior: the former carries the ball, the latter protects it. Moreover, depending on the actual environmental conditions, the roles can be swapped. Then, when the two robots are near the goal, the first robot exchanges the ball with the second robot, because the two robots have evaluated that the former can score more easily. The resulting complex emergent behavior (*exchangingBall*) seems to emphasize a deliberative aptitude of soccer robots to activate a cooperative cycle of actions. The sequence of actions reported in fig. 4, took place during the Challenge competitions, and was evaluated for the final Team score.

Conclusions

In this paper we have illustrated our current research, aimed to understand how much deliberative process should be endowed in the distributed control of a middle size team to play a soccer game cooperatively. Our current work evolves from our past experience to design behavior arbitration which triggers and it is triggered on the basis of purely *stigmergic* mechanism, namely, implicit coordination. Considering the inherent difficulty to force coordination when the dynamics of the game is not quite fast, we have tried to drive a cooperative task by a dynamical role assignment, switching from the implicit team assessment, given by the evaluation of quality functions, to the explicit negotiation on the *first notified/first advocated* basis. The actual implementation has been made possible with the heavy collaboration of the students at the Eng. School of Padua University who participate to the Research Project on RoboCup at IAS-Laboratory.

References

1. M.A. Arbib. Schema theory. In M.A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*, pages 830–834. MIT Press, Cambridge, 1995.
2. M.A. Arbib. Perceptual structures and distributed motor control. In *Handbook of Physiology-The Nervous System II: Motor Control*, pages 1449–1480. 1981.
3. R.C. Arkin. Motor schemas based navigation for a mobile robot. In *IEEE Conference on Robotics and Automation*, pages 264–271. 1987.
4. R.C. Arkin. Modeling neural function at the schema level. In *Biological Neural Networks in Inverterbrate Neuroethology and Robotics*, pages 383–410. 1993.
5. R. Brooks. A robust layered control system for a mobile robot. *IEEE Jour. of Robotics and Automation*, 2(1):14–23, 1986.
6. A. D'Angelo, E. Pagello, F. Montesello, and C. Ferrari. Implicit coordination in a multi-agent system using a behaviour-based approach. In *Distributed Autonomous Robot Systems 4 (DARS98)*, Karlsruhe (D), 1998.
7. E. Pagello A. D'Angelo F. Montesello F. Garelli C. Ferrari. Cooperative behaviors in multi-robot systems through implicit communication. *Robotics and Autonomous Systems*, 29(1):65–77, 1999.
8. M. Asada A. Birk E. Pagello M. Fujita I. Noda S. Tadokoro D. Duhaut P. Stone M. Veloso T. Balch H. Kitano and B. Thomas. Progress in robocup soccer research in 2000. In *Proceedings of the 2000 International Symposium on Experimental Robotics*. Honolulu, Dec 2000.
9. D. Kurabayashi. Toward realization of collective intelligence and emergent robotics. In *IEEE Int. Conf. on Sys., Man and Cyb. (IEEE/SMC)*, pages 748–753. Tokyo, 1999.
10. M. Mataric. Issues and approaches in the design of collective autonomous agents. *Robotics and Autonomous Systems*, 16(2-4):321–331, 1995.
11. G. Dudek M. Jenkin E. Milios and D. Wilkes. A taxonomy for swarm robotics. *Autonomous Robots*, 3(4):375–397, 1996.
12. E. Pagello C. Ferrari A. D'Angelo F. Montesello. Intelligent multirobot systems performing cooperative tasks. In *Special Session on Emergent Systems - Challenge for New System Paradigm, IEEE/SMC*, pages 754–760. Tokyo, 1999.
13. E. Pagello, A. D'Angelo, and E. Menegatti. Artisti veneti 2002 evolving an heterogeneous robot team for the middle-size league. In A. Birke and G. Kamilka, editors, *RoboCup2002*, Fukuoka (J), June 2002.
14. E. Pagello and A. D'Angelo et al. Emergent behaviors of a robot team performing cooperative tasks. *Advanced Robotics*, 17(1):3–19, 2003.
15. L. Parker. From social animals to teams of cooperating robots. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'97)*, Workshop on Multi-robot cooperation: Current trends and key issues. Grenoble (France), 1997.
16. G. Adorni A. Bonarini G. Clemente D. Nardi E. Pagello M. Piaggio. Art'00 - azzurra robot team for the year 2000. In P. Stone T. Balch and G.Kraetszchmar, editors, *RoboCup-2000: Robot Soccer World Cup IV*, number 2019 in Lecture Notes in Artificial Intelligence. Springer, Berlin, 2001.