22

Multi-Robot Systems and Distributed Intelligence: The ETHNOS Approach to Heterogeneity

Antonio Sgorbissa
University of Genova
Italy

1. Introduction

In the summer of 1999 the Azzurra Robot Team wins the second prize in the RoboCup competition in Stockholm. In the semi-finals, ART defeats the illustrious CS Freiburg (world champion in Paris 1998, Melbourne 2000, and Seattle 2001) in an unforgettable match that almost causes a heart attack in the supporters of either team. However, in the finals, ART is in difficulty against the very strong and quick team from Sharif CE University, Iran. Sharif players, which have been defeated by ART in the eliminatory rounds, seem to have learnt the lesson. Taking advantage of their undoubted superiority in mechanics, and the robustness of their vision and control algorithms (which, incredibly, run in MSDOS, the OS image being loaded from a bootable disquette), Sharif players are able to easily dribble ART players and to score three times before the end of the match, thus winning the first prize with a final score of 3-1.

In contrast with this preamble, RoboCup is not the subject of the discussion here: ART is rather taken as a case study of a "successful" Multi Robot system with very unique characteristics. This is perfectly in accordance with the declared purposes of RoboCup; that is, favouring scientific and technological advancements in robotics and embedded AI with the final aim of transferring the achieved results to real-world applications (Kitano et al., 1998). The choice of robotic soccer as a test field for the design of "autonomous agents" is justified by the intuition that – in most real-world scenarios –a step ahead towards Multi Robot systems is fundamental, if one wants to address issues such as efficiency, fault tolerance, quick response to user requests, or simply consider tasks which cannot be achieved by a single robot. Some examples are autonomous surveillance, rescue operations in large-scale disasters such as earthquakes or fires, humanitarian demining, cooperative manipulation or transportations of objects and furniture, autonomous explorations on desert areas, abandoned mines, or remote planets. To fully take benefit of the larger number of available resources, coordination is important if not essential; the underlying assumption in RoboCup is that these issues can be in part investigated by trying to coordinate a team of soccer-playing robots.

In this context, ART uniqueness is not due to the robots' perceptual skills, the mechanisms adopted for behaviour selection and execution, their coordination, localization, or planning capabilities. Its peculiarity is rather the extreme heterogeneity of the team itself: differently from the usual approach, in which a team is designed and built within a single University or

Source: Mobile Robots, Moving Intelligence, ISBN: 3-86611-284-X, Edited by Jonas Buchli, pp. 576, ARS/plV, Germany, December 2006

research laboratory, ART has been the only "National" team in the history of RoboCup, being composed of different robotic players developed in six Italian universities autonomously from each other (Rome – team leader, Genova, Milano, Padova, Palermo, Parma). This gave birth to players which were very different both in the hardware and software architecture (see Fig. 1), and was further accentuated by the fact that research groups involved in ART agreed only partially (and sometimes had very passionate debates) on the philosophy according to which autonomous agents should be designed and built. Heterogeneity in Multi Robot systems have been widely investigated in literature, and journal special issues have been specifically dedicated to the subject (an attempt to formalize behavioural heterogeneity can be found in Balch, 2000); the main difference is that, in ART, heterogeneity was not a choice to develop autonomous agents with specialized functionalities, but rather a very strong constraint to deal with. In spite of this, after training together for less than one week before RoboCup, and some minor changes in the onboard software and hardware, the robots managed to successfully cooperate and achieve a significant result.

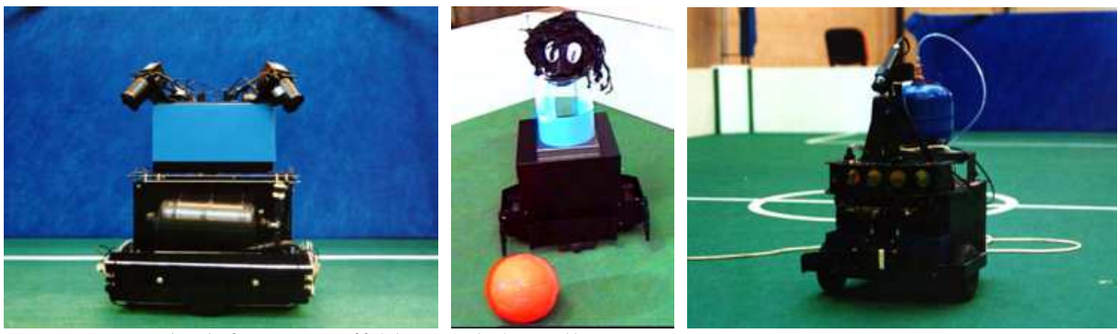

Fig. 1. From the left: TinoZoff (the goalie), Rullit, Homer.

In the chapter, I argue that one of the main responsible of ART success in 1999 was the middleware adopted by the research units as a starting platform to develop robotic players: ETHNOS, a Real Time extension to the standard Linux kernel which provides a general architectural framework for the design of distributed robotic applications. When comparing it with other architectural approaches, the most tangible difference consists in the fact that ETHNOS does not make any cognitive assumption (e.g., behaviour based, hybrid deliberative-reactive, etc.). In fact, it should be more likely considered as an "Operating System" for mobile robotics; even if, literally, it is not an OS since it is implemented on top of the standard Posix API, the founding idea is to identify common needs and characteristics in mobile robotic systems, and to provide "ETHNOS system calls" to decompose their inherent complexity without annoying the developer with a pre-established cognitive architecture. As a side effect of its versatility, students from different Universities involved in ART, even when writing their own algorithms in complete autarchy and in total contrast with the healthy principle of code reuse, started nevertheless to talk a common language thanks to ETHNOS; an important achievement, if one considers the educational objectives of RoboCup. Even more important, after 1999 many Italian research groups continued to use the system, both for RoboCup and in different scenarios: among the others the Golem team, which won the second prize in Melbourne 2000, and was completely designed by undergraduate students autonomously from the University (De Pascalis et al., 2001).
Nowadays many programming frameworks for robotics share a similar philosophy. In spite of this, ETHNOS still contains original elements: in the next Sections, these issues will be addressed in depth. First, Multi Robot systems in literature will be reviewed under different perspectives;

out of the hundreds of article on the subject, we select a representative subset for sake of brevity. Second, ETHNOS will be described, by outlining its relevance in determining the success of the Azzurra Robot Team in RoboCup99; design choices will be justified through theoretical investigation and experimental validation. The last part of the chapter will address our recent research, where the ETHNOS multi-robot paradigm is extended in order to include a broader conception of Multi Robot and Multi Agent coordination: the deployment of robot teams in intelligent environments, where they cooperate with a network of fixed intelligent devices for the achievement of common objectives.

2. Related Work

In the last years, Multi-Robot systems are receiving an increased attention in the scientific community. In part, this is due to the fact that many problems related to the control of a single robot have been – at least partially – solved, and researchers start to look at the massive introduction of mobile systems in real-world domains. In this perspective, Multi Robot systems are an obvious choice for all those applications which implicitly take benefit of redundancy; i.e., applications in which, even in absence of a coordination strategy, having more robots working in parallel improves the system's fault tolerance, reduces the time required to execute tasks, guarantees a higher service availability and a quicker response to user requests. There are many cases for which this holds: autonomous cleaning (Jung & Zelinski, 2000; Kurazume et al., 2000); tour guiding in museums, art-galleries, or exhibitions (Siegwart et al., 2003); surveillance and patrolling (Rybski et al., 2002; Vidal et al., 2002), rescue in large scale disaster such as fires, floods, earthquakes (Bahadori et al., 2005); landmine detection (Acar et al., 2003); autonomous exploration and mapping, possibly with the additional constraint of retrieving samples or objects of interests (Rekleitis et al., 2000; Burgard et al., 2005). All the previous applications share some common characteristics: they require the execution of a set of loosely coupled tasks more or less comparable to "coverage", i.e., the problem of visiting the largest number of locations in a given area (Fontan & Matarić, 1996; Choset, 2001), and "foraging", i.e., the problem of finding objects or persons of interest ("food", in the metaphor) and returning them to a specified location (Arkin, 1992; Drgoul & Ferber, 1993). If we ignore issues related to the interference between robots operating in the same area (Fontan & Matarić, 1996; Gustafson & Gustafson, 2006), coverage and foraging obviously improve, when more robots operate in parallel, even in absence of coordination; however, coordination through implicit or explicit communication can provide a significant aid (Balch & Arkin, 1994).

When taking into account more complex scenarios, Multi Robot systems are no more an option; consider, for example, a team of robots trying to achieve an objective which cannot be attained by a single robot, or maintaining a constant spatial relationship between each other in order to execute a task more effectively. Examples of the first type are cooperative manipulation and transportation (Berman et al., 2003; Yamashita et al., 2003), pushing (Matarić et al., 1995; Parker, 1998), or other tightly coupled tasks; examples of the second type are moving in formation or in a convoy for patrolling or transportation, a strategy which allow team members to improve security by protecting each other against attacks, or supporting each other in a different way (Balch & Arkin, 1998; Beard et al., 2001; Barfoot & Clark, 2004). In these and similar scenarios, coordination becomes fundamental, introducing new challenges and opening new possibilities (Balch & Parker, 2002). One of the first issues to be faced is whether coordination should be formalized and solved through a centralized

control process, which computes outputs to be fed to actuators to perform a coordinated motion of all the components of the system, or distributed, with each member of the team taking decisions in autonomy on the basis of its own sensorial inputs, its internal state and – if available – information exchanged with other robots (for sake of brevity, we purposely include in the same class "decentralized" and "distributed" approaches). Whereas central approaches often lead to an optimal solution, at least in a statistical sense, distributed approaches usually lack the necessary information to optimally solve planning & control problems. They are nevertheless more efficient to deal with real-word, non structured scenarios: since decisions are taken locally, the system is still able to work when communication is not available, and – in general – allow a quicker, real-time response and a higher fault-tolerance in a dynamically changing environment. Most teams in RoboCup share these considerations, even if counterexamples exist (Weigel et al., 2002). Notice also that, a part from optimality, univocally accepted metrics or criteria to compare the performance of centralized and distributed Multi Robot systems are missing; the problem is raised, for example, in (Schneider, 2005).

The dual problem is cooperative sensing, in which robots share their perceptual information in order to build a more complete and reliable representation of the environment and the system state. The most notable example is collaborative localization and mapping (CLAM, see Fox et al., 2000; Schmitt et al., 2002; Madhavan et al., 2004), which has been recently formalized in the same statistical framework as simultaneous localization and mapping (SLAM), as an extension of the single robot problem. However, the general idea is older and different approaches exist in literature, often relying on a sharp division of roles within the team (Kurazume et al., 2000). Cooperative sensing can include tracking targets with multiple observers (Parker, 2002; Werger & Matarić, 2000), helping each other in reaching target locations (Sgorbissa & Arkin, 2003), or the pursuit of an evader with multiple pursuers (Vidal et al., 2002). Notice also that cooperative sensing often relies on cooperative motion, and therefore they cannot be considered as totally disjoint classes of problems.

When considering Multi Robot coordination at a higher level of abstraction, architectural and organizational issues become of primary importance. In (Broten et al., 2006), software architectures for robotics are classified as "reference" or "emergent" architectures. Reference architectures usually implement a cognitive architecture, and therefore pose very tight constraints on how a complex system should be decomposed into interacting components; depending on the cognitive assumptions made, these can be – from time to time – behaviours, motor schema, functional blocks, agents of different nature, etc. Reference architectures can be more targeted to single robot systems (Brooks, 1986; Bonasso et al., 1997; Konolige & Myers, 1998), Multi Robot (Parker, 1998; Pirjanian et al., 2000; Gerkey & Matarić, 2002) or both (Arkin, 1990; Simmons et al., 2002). When specifically designed for Multi Robot systems, reference architectures address such issues as the allocation of tasks to multiple robots, each possibly having different hardware and software capabilities, dynamic role assignment to perform a joint task (Parker, 1998; Gerkey & Matarić, 2002; Stone & Veloso, 1999), route planning with multiple vehicles and goals (Brumitt et al. 2001), etc. Indexes considered to evaluate an approach are near-optimality of the assignment, stability of the solution in presence of fluctuations in the environmental conditions, smooth performance degradation when resources are temporarily unavailable, robustness in presence of uncertain and incomplete information. Behaviour-based, auction based, or similar distributed approaches (with or without explicit communication) are very common; this happens for the reasons already

discussed, not last the fact that optimality in task and role assignment is computationally very expensive to achieve, and consequently inappropriate for real-time operation. Emergent architectures, on the opposite, do not make strong assumptions on cognitive aspects. They rather specify the internal organization and the interactions between software processes, providing a framework to decompose complexity into simpler software components; for this reason, they are also referred to as "Middleware" or "Software Frameworks" for robotics. Services provided range from inter-process communications, support for code reusability and fast-prototyping, hardware simulation. Examples are Player/Stage (Gerkey et al., 2003), MIRO (Utz et al., 2002), Marie (Cotè et al., 2006), CLARATy (Nesnas et al. 2006), SmartSoft (Schlegel 2006). A joint effort in this sense is the OROCOS project: www.orocos.org), whereas code reusability and portability in robotics is especially addressed in (Alami et al., 2000).

RoboCup offers many opportunities to investigate all the issues discussed so far. Synchronised actions, such as passing the ball to each other until a goal is finally attempted, require the robots to execute tightly coupled tasks; Multi Robot formations can be considered - as in human play – as a mean to improve the effectiveness of game strategies, enabling robots to support each other when attacking or defending their own goal; the new trend in collaborative localization and sensing has been significantly speeded-up by research in RoboCup, as well as task/role allocation strategies and algorithms according to different architectural approaches. In this general context, we mostly address architectural aspects, i.e. we investigate a software architecture for:

- Decomposing the complexity of a single robot player into basic software components, providing services for real-time scheduling and analysis, real-time inter-process communications, code reusability.
- Decomposing the complexity of a Multi Robot system into a distributed team of autonomous robots, providing support for inter-robot communication in a real-time scheduling context, and addressing specific issues such as real-time task allocation and role assignment in a distributed fashion.

Under the following constraints:

- The software architecture must adapt to a team of heterogeneous robots which differ significantly both in the hardware and in the software, allowing to implement different cognitive architectures according to the approaches adopted by the various research units (hence it is an "emergent" architecture).
- Inter-robot communication and coordination protocols must be robust to changing environmental conditions in a very dynamic and "hostile" environment, allowing smooth performance degradation in the case that inter-robot communication or robots themselves are temporarily unavailable.

Cooperative motion and sensing are not explicitly addressed; however, some architectural choices determine the appearance of unforeseen cooperative behaviours, even if in a rudimental form. On one side, this is nothing more but another point in favour of the claim that intelligent behaviour is an emergent property; the interesting result is that, in the case of ETHNOS, intelligence seems to emerge from OS design choices, even in absence of cognitive assumptions on the nature of ongoing processes, their interaction with the environment, and their mutual interactions. Whether this means that every architecture is inherently "cognitive", is a question that would deserve a deeper discussion. In the following, we limit ourselves to discuss ETHNOS in details, by leaving the reader to draw her own conclusions.

3. ETHNOS

ETHNOS (Expert Tribe in a Hybrid Network Operating System[1]) is a framework for the development of Multi Robot systems. It is composed of:

- A dedicated distributed real-time operating system, supporting different communication, execution, and representation requirements; these functionalities are built on top of a Posix compliant Linux kernel.
- A dedicated network communication protocol designed both for single robot and Multi Robot systems, taking noisy wireless communication into account.
- An object oriented Application Programming Interface (API) based on the C++ language (and a subset based on Java).
- Additional development tools (a simulator, a Java-applet template, etc.).

The reference architecture of a single robot, and consequently of the ETHNOS operating system, is entirely based on the concept of "experts", concurrent agents responsible for specific activities. Experts in ETHNOS can be organised in groups, possibly distributed in a computer network, depending on their computational characteristics: the type of cognitive activity carried out, duration, type of data managed, etc. However these groups do not have a hierarchical organisation but, on the contrary, are executed in a flat model in such a way that the real-time constraints are not violated. This generality in the expert specification allows the use of ETHNOS with different architectural choices without loosing real-time analysis, execution and inter-robot communication support. This aspect has been very important in ART, in which:

- Bart and Homer robots (Padova) relied on a behaviour-based model with arbitration.
- Relè (Genova) adopted a hybrid deliberative/reactive model.
- Rullit robot (Milan) was based on fuzzy logic control.
- TinoZoff goalkeeper (Parma) was based on a central controller.
- TotTino and RonalTino (Rome) were developed in the Saphira architecture, and rely on ETHNOS only for inter-robot communication and coordination.

The next paragraphs will deal with these properties in greater detail.

3.1 Inter-Expert and Inter-Robot Communication

ETHNOS allows the robots to be programmed in such a way that the different experts can be integrated, during development but also at run time, with little or no intervention in the software of the expert itself, thus facilitating both rapid prototyping and dynamic architectural reconfiguration. The first property allows the development of a robotic application, from a set of basic components or behaviours, even by non-highly specialised programmers (for industrial purposes but also for didactic activity in student projects, particularly relevant in RoboCup). The second property allows the system to switch at run-time from a configuration in which it is performing a certain task (for example in which the robot is attacking and thus the active experts are responsible of ball pushing, path planning, and obstacle avoidance) to a different configuration (for example in which the robot is defending and the active experts are responsible of opponent-tracking, ball interception, etc.). These properties are achieved by exploiting a suitable inter-expert communication protocol which comprises of three different

[1]In ETHNOS the expression "expert tribe" draws an analogy between a robotic system and a tribe composed of many experts (concurrent agents) in different fields (perception, actuation, planning, etc.). The term "hybrid" indicates the support to the integration of cognitive agents of different nature: deliberative, reactive, subsymbolic, etc; the term "network" refers to the possibility of distributing components on a network. For documentation refer to http://www.ethnos.laboratorium.dist.unige.it.

communication methods (a comparison between communication patterns and their relevance in component-based robotics can be found in Schlegel, 2006):

- Asynchronous message-based communication.
- Synchronous access to an expert (similar to function calling).
- Access to a shared representation.

The first method is the most general of the three and it is at the base of ETHNOS applications, hence it is the only one discussed here. It is a message-based communication protocol (EIEP – Expert Information Exchange Protocol) fully integrated with the expert scheduler. EIEP encourages the system developer to de-couple the different experts in execution to reach the limit situation in which the single expert is not aware of what and how many other experts are running; in this way, an expert can be added or removed at run-time without altering the other components of the system. Expert de-coupling is achieved by eliminating any static communication link: EIEP is an asynchronous message-based method in which the single expert "subscribes" to the desired message types, known a priori. When another expert "publishes" any message of the subscribed types, the subscriber receives it transparently. In this way, an expert does not know, explicitly, who the sender of a message is or to which other experts, if any, its message is being distributed. Moreover, the same principles apply also if planning & control activities are distributed in a computer network (for example, to deal separately with on-board reactive components and remote high-level components responsible of computationally intensive tasks: Fig. 2 on the left).

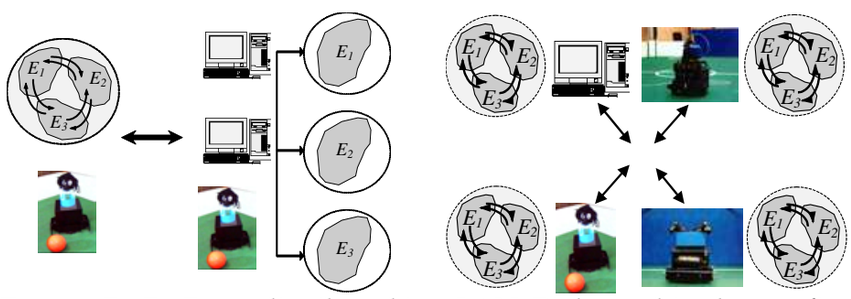

Fig. 2. Left: Experts E_1, E_2, E_3 are distributed in a LAN. Right: multi-robot configuration.

EIEP provides also transparent inter-robot communication support. Using the same publish/subscribe principle, messages can be exchanged on the basis of their content not only within the same robot but also among different robots (e.g., players within the same team). However, in inter-robot communication, it is often important to distinguish between internal messages (messages to be distributed within the experts implementing a single player) and external messages (messages to be sent from a player to another) to avoid the explosion of unnecessary network data communication (e.g., commands to be issued to actuators or sonar readings are meaningful only locally). In ETHNOS the different experts are allowed to subscribe to "communication clubs"; we may envisage a single club to which the different players belong or even different clubs, to which only players which are performing a cooperative activity belong, for example the one which carries the ball toward the goal and the one which is supporting it. Message subscription and publication can thus be distinguished in internal (possibly distributed), internal and external in a specific club, or internal and external in all clubs. Again, it is the responsibility of the system to transparently distribute the messages to the appropriate receivers; Fig. 2 on the right shows robots that communicate in a single club, to which all of them have subscribed together with an external supervisor. Because of EIEP, whenever we want to add or remove a player, it is not necessary to explicitly inform other

players about the modifications in the team's composition. This has been very important in ART in which there were more than four types of robots available, with different characteristics of play, and thus different solutions were selected for the single matches and also modified during the matches to better contrast the opponent.
Finally, in wireless communication, transmission packets are sometimes lost due to noise or interference. In this context, both TCP/IP and UDP/IP cannot be used: the former because it is intrinsically not efficient in a noisy environment; the latter because it does not guarantee the arrival of a message, nor provides any information on whether it has arrived or not (for efficiency reasons, ETHNOS communication is built on top of the standard Unix Socket interface). To deal with this, ETHNOS comprises a protocol called EEUDP (Ethnos Extended UDP). EEUDP allows the transmission of messages with different priorities. The minimum priority corresponds to the basic UDP (there is no guarantee on the message arrival) and should be used for data of little importance or data that is frequently updated (for example the robot position in the environment, that is periodically published). The maximum priority is similar to TCP because the message is sent until its reception is acknowledged. However, it differs because it does not guarantee that the order of arrival of the different messages is identical to the order in which they have been sent (irrelevant in ETHNOS applications because every message is independent of the others), which is the cause of the major TCP overhead. Different in-between priorities allow the re-transmission of a message until its reception is acknowledged for different time periods (i.e. 5 ms, 10 ms, etc.). Notice that, when sending messages with the minimum priority, EEUDP is an efficient implementation of MailBoxes in a distributed scenario, an asynchronous communication method particularly suited to real-time applications (Buttazzo, 1997) since it does not delay the execution of hard-real tasks (even if Ethernet-based communication – in a strict sense – is not real-time, since the time required for packet-delivering is not predictable).

3.2. Real-Time Expert Scheduling

ETHNOS supports real-time analysis and execution of experts. Throughout the chapter, the term real-time is used in its formal sense, by referring to the mathematical theory of real-time scheduling and communication as depicted, for example, in (Buttazzo, 1997). This is not a common approach in mobile robotic architectures (with some exception, e.g. Musliner et al., 1993; Rybsky et al., 2003); however, we argue that schedulability analyses assume primary importance whenever computational resources are limited, which is exactly the case for RoboCup (and all other systems which have constraints on their dimensions, weight, an on-board power supply). Suppose that one needs to understand the cause of an observed anomalous behaviour: e.g., the robot stands still whereas we would expect it to move towards the ball. Where does this behaviour come from? Is it a bug in the code, or a temporary CPU overload that delays time critical tasks? Verifying that each task, when taken singularly, meets deadlines is not sufficient; real-time scheduling analyses allow to consider the set of tasks as a whole, and to know in advance if what we are asking the CPU to do is feasible at all.
Before discussing ETHNOS scheduling policies, we qualitatively analyse the timing requirements of RoboCup players. Results can be summarized as follows:

- Sensor acquisition is handled by periodic tasks. They are executed at a high frequency, their computational time is usually low and predictable: e.g., reading encoders; reading sonar or laser rangefinders; acquiring video, etc.
- Reactive planning & control activities are periodic as well. Frequency is high, computational time is low and predictable: e.g., computing speed & jog values for reaching or pushing the ball; avoiding obstacles; updating odometry, etc.

- Data filtering, segmentation, and aggregation activities are periodic or sporadic (their period depends on sensor acquisition) with a computational time at least one order of magnitude greater, but still predictable: e.g., detection of color blobs; building maps; localization of important features, etc.
- Inter-robot communication is aperiodic, with a non-predictable computational time; in a distributed context, a robot does not know in advance when and for how long it will receive messages from teammates.
- Aperiodic, computationally intensive activities can be optionally considered, whose time cannot be easily predicted but – usually – is orders of magnitude greater: collaborative self-localization, symbolic planning, etc.

Notice that only the former four activities are critical for the system, whereas the latter class includes activities that increase performance, but are not critical. An overall architecture for the development of RoboCup players must take these aspects into account and thus permit the integration of less computationally intensive activities (whose execution is critical for the system, and therefore have hard real-time requirements) with more computational intensive ones (whose execution is not critical for the system, and therefore have not real-time requirements).

Periodic, time bounded, critical activities (Sensor acquisition, Reactive planning & control, Data filtering, segmentation, and aggregation) are executed in high priority experts according to two possible algorithms: Rate Monotonic or the non-preemptive Earliest Deadline First algorithm (Jeffay et al., 1991). EIEP allows exchanging information between real-time experts asynchronously, thus avoiding unpredictable delays or priority inversion phenomena. Aperiodic, not time-bounded, non-critical activities are executed in low priority experts; they run in the background and communicate their results to real-time experts whenever their outputs can improve the behaviour of the system. Asynchronous communication through EIEP guarantees that background experts do not interfere with the real-time scheduling policy of the system. Inter-robot communication deserves a special attention. In fact, if we want the robot to coordinate through explicit communication, this should be considered a critical activity; however, it cannot be scheduled as the other real-time tasks since we are not allowed to make reasonable assumptions about when and for how long messages are going to be exchanged with other robots. The solution adopted is common in literature for aperiodic real-time tasks: we insert in the system a couple of Periodic Servers (Lehoczky et al., 1992), i.e. real-time periodic tasks which are given the lowest period (hence the minimum latency) and a pre-defined "capacity" to serve requests from aperiodic tasks. In our case, the capacity of RxServer and TxServer is spent, respectively, for receiving and sending messages. The mechanism guarantees that a predefined amount of CPU time is always devoted to inter-robot communication, thus avoiding starvation. Taking all these consideration into account, ETHNOS implements different possible scheduling policies ranging from two extremes (Fig. 3) in which:

- Real time Experts (periodic, sporadic, and Periodic Servers) are mapped, one to one, into different Posix threads and scheduled as stated by the Rate Monotonic algorithm, whereas background experts are assigned a lower priority and scheduled in time-sharing.
- All real time Experts are mapped into a single high priority thread, which non pre-emptively schedules the real-time as stated by the non-preemptive Earlies Deadline First algorithm, whereas background experts are assigned a lower priority and scheduled in time-sharing.

The two extreme solutions (as well as intermediate ones) can be adopted depending on the application considered. In the former, the Rate Monotonic algorithm (chosen because of its ease of implementation in every Posix compliant OS) is exploited, allowing low latencies in expert execution (depending on the granularity of the Posix scheduler). In the latter, low latency is traded off with an increased processor utilisation, since the presence of only one thread for high frequency tasks guarantees a lower context switching rate with respect to a fully pre-emptive scheduler; moreover, it allows to access critical regions without synchronization primitives and related problems.

As well as providing transparent real-time execution (automatic computation of the required priorities) ETHNOS also includes tools for analysing the feasibility of the scheduling. To allow this, in a preliminary phase, the worst execution time for each expert is computed, and continuously updated when the system is running and also across different runs. Whenever the user asks the system to schedule a set of experts with the selected RM or EDF algorithms, ETHNOS acts as follows (Fig. 3 on the right):

- For each expert (periodic or sporadic) the scheduling conditions (for RM or EDF) are tested referring to its worst execution time and the period (for sporadic experts, the Minimum Interrarival Time between two consequent runs). The "capacity" of RxServer and TxServer is automatically computed.
- If the scheduling is possible, experts are scheduled according to the selected policy. Periodic experts are scheduled according to their period, whereas sporadic and aperiodic ones are activated whenever they receive a EIEP message to which they have subscribed.
- Should an expert miss its deadline (this is possible because of the approximate execution time analysis) a very simple recovery policy is adopted: the next deadline of the expert is obtained by adding the expert's period to the end of the last run (not to the end of the previous period). This is equivalent to changing the phase of the expert, saving the processor from computational overload and from the consequent deadline missing by other experts.
- Background experts are scheduled when no other expert is ready for execution. Should we need a temporary intensive activity in background it is a matter of the specific application to reduce real-time tasks activities.

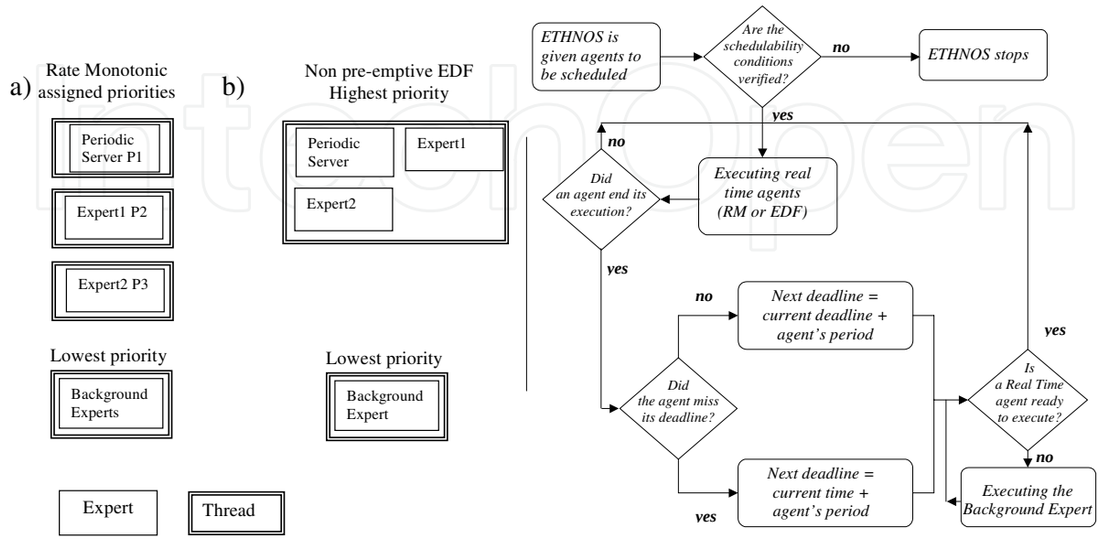

Fig. 3. Left: Mapping experts onto threads with RM and EDF. Right: ETHNOS scheduler.

Some technical considerations: recently, programming frameworks have been proposed relying on Linux RT extensions (e.g., RTAI – www.rtai.org) to exhibit hard real-time characteristics. On the opposite, we have chosen the Linux standard distribution, which follows Posix specifications for real-time (thus being very portable), but have many limitations: in particular, does not allow scheduling tasks at very high frequency (>100Hz) and has significant jitters in task execution. Our choice is justified by the following two considerations. First, most hardware platforms for mobile robotics have dedicated processors or microcontrollers for low-level I/O functions that are executed at a high frequency: the tasks running on the on-board computer have looser requirements (i.e., frequency is << 100Hz), and are therefore compatible with the standard Linux kernel capabilities. In our view, it is most important to deal with limitations in computational resources, than guaranteeing latencies below 1ms. Second, Posix compliance guarantees portability, and allows us to easily recompile a new version of ETHNOS as soon as a new version of the underlying Linux kernel is available. ETHNOS currently takes benefit of the 2.6 series of the Linux kernel, which is considerably improved and therefore more adequate to real-time applications: among the others, it is now fully re-entrant, the priority-based internal scheduler works in constant time, and its granularity is one order of magnitude smaller (1ms). An RTAI-enhanced version of ETHNOS designed for the control of hybrid systems, composed of mobile robots and manipulators, has been recently made available.

3.3. Coordination: Explicit Role allocation

Efficient inter-robot communication and real-time scheduling are the building blocks of Multi-Robot coordination in ART (Iocchi et al, 2003), which has the main purpose of

- Selecting the specific activity that each robot has to execute (e.g., attacking, defending, etc.), according to the current situation (i.e., location of the ball and of the opponents) and the currently selected game strategy.
- Adequately distributing players and on the field.

The previous two concepts are synthesized in the concept of "role": roles imply both a subset of activities that the robot is asked to execute, and a territorial division of the playing field. Game strategy is achieved through the concept of "formation"; formations determine which roles, among a set of available ones, must be active in the current situation, and the percentage of robots to be assigned to each active role. Coordination in ART is not oriented to tightly coupled tasks, such as passing the ball to each other, or similar; more similarly to "coverage" and "foraging", the underlying principle is that of taking benefit of redundancy, by reducing interference on the basis of an efficient division of tasks and competencies. Consider the role "striker": it includes a subset of activities aimed toward reaching the ball, dribbling the opponents and the goalie, and finally score (details on how this is done depend on each player's hardware and software). The role "striker", as well as other roles (e.g., "defender", "supporting striker", etc.) must be assigned to the robot which, given the present situation and its intrinsic capabilities, is most suited to it. In this general scenario, and by assuming that inter-robot communication is available, we deal with the following additional requirements and assumptions:

- Smooth degradation: in temporary absence of communication, the robots must be able to perform sub-optimally, on the basis of the information available.
- Heterogeneity: the robots significantly differ in their hardware and software, i.e. they can usually perform the same activities, but in a different way and with different performance.
- Stability: since robots operate in a highly dynamic and hostile environment, roles must be periodically re-assigned; however, role-assignments must be stable in

presence of small fluctuations in the input parameters, to avoid frequent and unproductive changes in robots behaviour.

The strategy adopted in ART is based on the concept of an utility function $f_{i,r}(\ldots)$ which returns, for every robot i, a subjective evaluation about how much the robot "feels confident" in assuming role r. If the robot is able to compute $f_{i,r}(\ldots)$ for every role which is active in the current formation, and to broadcast the value to other players, a very efficient algorithm for distributed role assignment can be implemented. That is, given:

- A set T composed of n robots t_i.
- A set R composed of m roles r_j which are active in the current formation and ordered according to their priority, i.e., assigning r_j is more important than assigning r_{j+1}.
- A ratio p_j of "required covering" for each role ($0 \leq p_j \leq 1$), which determines the number $n_j = ceil(p_j \cdot n)$ of robots in the team which must assume role r_j.

Each robot T_i in T acts as follows:

- For each role r_j in R: compute and broadcast $f_{i,j}$.
- For each robot $Tk \neq i$ in T, and for each role r_j: collect the broadcasted values $f_{k,j}$.

At this point, each robot t_i knows how much each robot in the team (included itself) feels confident in assuming role r_j . Next, t_i computes role-robot assignments:

- For each role r_j in R, and for c=1 to n_j : assign the role r_j to the robot t_l in T for which $l = argmax_k(f_{k,r})$; delete t_l from T (to prevent a robot to take many roles).

This obviously requires each robot to estimate how adequate it is to perform a given activity. This is a very critical point; in (Iocchi et al, 2003) details are provided on how $f_{i,j}(\ldots)$ is modelled for each role and each robot, depending on its hardware and software characteristics, through an experimental phase in which all the parameters involved must be carefully tuned. Omitting details, $f_{i,j}(\ldots)$ is a weighted linear combination of different components, taking into account the variables of the robot state that are considered relevant for each role. Examples are distance from the ball; maximum speed achievable; presence of opponents or teammates in the neighbourhoods; intrinsic sensing or motion capabilities; external or internal causes that can obstruct motion. Each component heuristically measures how close the robot is to achieving the objectives associated to the role; experimentation provides insight on their relative weights in the utility function.

The algorithm heavily relies on ETHNOS communication protocol: each robot t_i transmits updated utility values for each $f_{i,j}(\ldots)$ approximately every 100ms (i.e., the period of the RxServer and TxServer in the current implementation). EIEP do not require any static communication link to be established: in role-assignment, this allows each robot to autonomously adapt its behaviour to the number and the identity of robots currently in the field, even in presence of temporary malfunctioning or sudden substitutions in the team composition. In fact, from ETHNOS point of view:

- Every robot subscribes to the MSG_UTILIY_FUNCTION message type.
- Every 100ms, each robot publishes a message containing its utility functions.
- When a robot receives a message from another robot, it stores utility values. If a new message from the same sender is not received in the meanwhile, the last received utility values are kept alive for 1 second (this is important to avoid tightly synchronized communication and to deal with communication noise).
- Every 100ms each robot compares its utility values with those that it has stored in memory, and assigns roles according to the algorithm.

The system exhibits many good properties in a distributed, dynamic environment in which communication is not guaranteed. First, if all robots receive all the utility values broadcasted

by teammates, the task assignment is optimal and – obviously – identical for all robots. Computational complexity of the assignment algorithms is $O(mn)$, with n the number of robots and m the number of roles. (Parker 2000) demonstrates that task assignment for a generic team of heterogeneous robots is NP-hard. This apparent discrepancy derives from the fact that we are not trying to maximise a global evaluation function computed over the whole team; instead, we simply assign the role r_1 with the highest priority (out of an ordered set) to the $n_1=ceil(p_1 \cdot n)$ robots which are best suited to it, and iterate the procedure over all the set of available roles. Transmission complexity, which determines bandwidth requirements, is also $O(mn)$; since the number of roles is usually small and lower than the number of robots, the size of each broadcasted messages is small as well, and the bandwidth required is very low. Second, if some or all the robots are temporarily unable to communicate their values, the algorithm shows a very robust, sub-optimal behaviour: each robot proposes an assignment which takes in account only the robots with which it is in contact, considering the highest priority roles first. This leads to the limit situation in which, if communication is completely unavailable, each robot takes the highest priority role, which is very reasonable in most scenarios. Finally, since inter-robot communication is not perfectly synchronized, it can happen that, at a given instant, not all the robot propose the same role assignment. To limit the consequences of this, and to avoid oscillations whenever different robots return comparable "utility values" for the same role, a simple stabilizing mechanism is introduced; when computing the new utility values, each robot receives an additional score for the role that it is currently playing. This has the effect of eliminating small fluctuations through a sort of "hysteresis" which affects the computation of $f_{i,j}(\ldots)$: being assigned a given role is much harder than conserving it. A parallel with the influence of hysteresis in the assignment of social roles in human communities would be intriguing, but this is not definitely the place for it.

4. Experimental Results

The reliability of EIEP communication has been experimentally verified both in benchmarks and during RoboCup matches. One of ETHNOS peculiarity is the fact that, for network communication, the system allocates a maximum guaranteed and dedicated time (also referred to as capacity) within a couple of expert which are given the highest priority possible (i.e., RxServer and TxServer). The server capacity value is computed automatically on the basis of the schedulability analysis, in such a way that the real-time execution of other experts is also guaranteed; moreover, since servers has the maximum priority, we are guaranteed that, within a predictable and reasonable amount of time, all messages to and from other robots will be sent and received (without considering problems related to communication noise).

In Fig. 4, the four graphs represent different machines (with different processing power) corresponding to three robots (Relè, Homer and Bart) and a coach (a visual interface), connected using wireless Ethernet. The top line in the graph indicates the calculated time available every 100ms for communication purposes; clearly this value decreases as the number of experts in execution increase (and so the computational load). The bottom line indicates the time spent in communication every 100ms; since, for this experiment, we have assumed that the activity of every expert involves either transmission or reception of messages, this value increases with the number of experts. In this way it is always possible to determine a priori whether the system is capable of both communicating information and executing in real-time. For example, if we

consider Relè, the limit situations in which the two lines converge is also the limit beyond which the schedulability analysis fails. Instead, if we consider Bart, real-time scheduling is possible also beyond the intersecting point, but only if we accept communication degradation. Notice that, in real-time systems, an alternative and very common solution to deal with network communication is to execute it in low-priority, non real-time tasks (e.g., QnX, LinuxRT, RTAI). The problem is that, if real-time tasks heavily utilize the CPU, background communication activities can be delayed for an indefinite time, which is definitely not desirable.

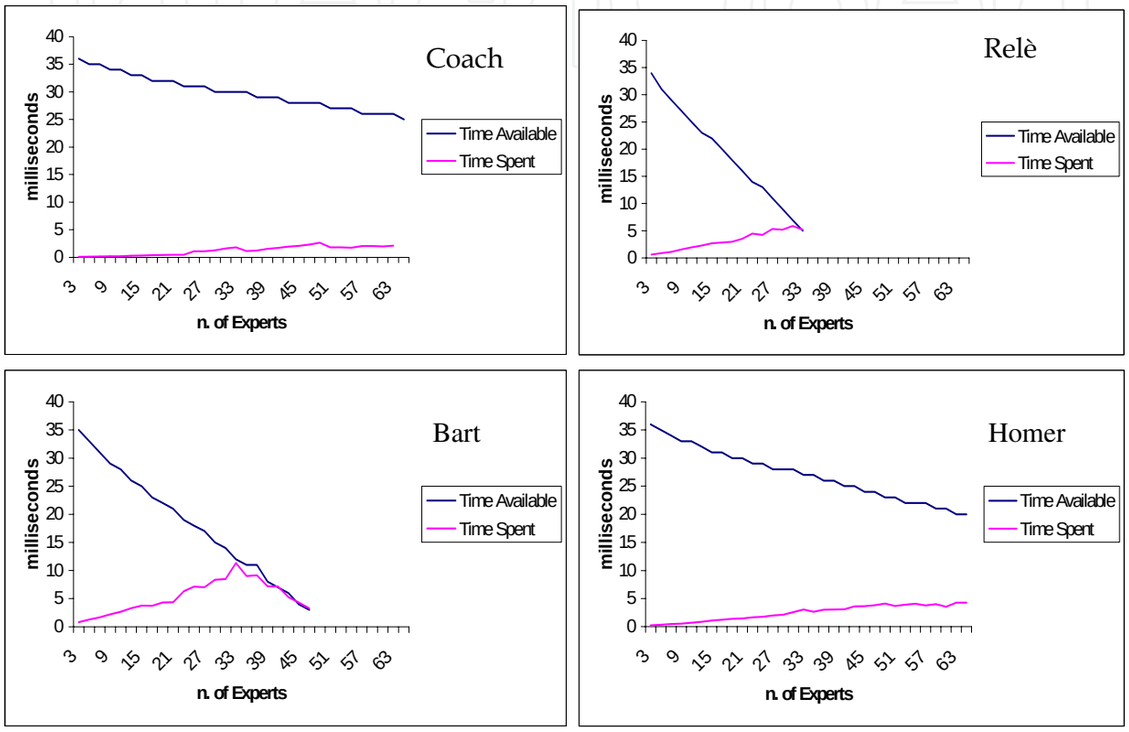

Fig. 4. Network Communication in ETHNOS.

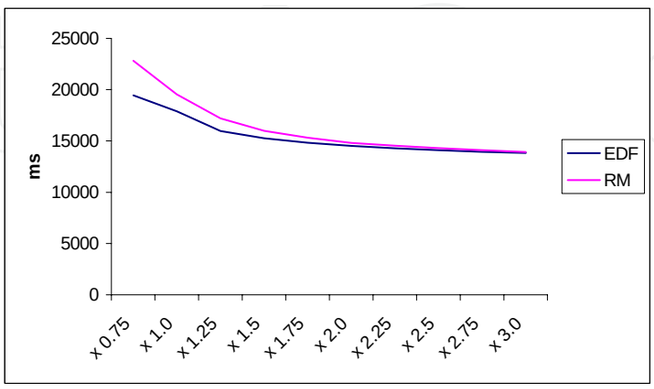

Fig. 5. Time taken to complete operation with different task periods.

To clarify the concept consider Fig. 5, which compares the two scheduling techniques RM and non-preemptive EDF available in ETHNOS. In the experiment, a fixed number of

experts with a pre-defined computational time are scheduled together with a single background expert; in absence of RxServer and Tx Server, the background expert can be devoted to network communication, or other computationally intensive activities (collaborative localization, planning, etc.). The time taken to carry out the background expert is plotted against the factor applied to the base period of the periodic experts; the smaller the factor, the shorter the periods and therefore the higher the processor load. As expected, when the processor load is high, the time required by the background expert to end its computations increases; moreover, the time available significantly depends on the context switches rate, as outlined by the fact that non-preemptive EDF is more efficient than RM in these conditions. As a consequence, communication in background turns out to be very hardly predictable.

Communication in ETHNOS is one the building blocks for distributed role assignment; however, it favours also the emergency of unforeseen coordinated behaviours. EIEP allows to specify whether a message must be delivered only internally, i.e., to experts belonging to the same robot, or externally, i.e., to all robots which belong to the same communication club. An example of the first type is MSG_SPEEDJOG, which is published by the expert which generates the trajectory but it is obviously delivered only to local motion experts. Examples of the second type are MSG_UTILITY_FUNCTION, or MSG_STARTSTOP, the latter used to notify all the robots that the match has started or has been suspended. In experiments performed during the one-week training phase before RoboCup, we programmed robots to share externally also messages of type MSG_BALL_PERCEPTION, which contain the perceived position of the ball in a fixed reference frame. Notice that, at that time, the information was meant to be used by a graphical monitor to visualize the current situation in the field for debugging purposes; during RoboCup, there were attempts to use the same mechanism to fuse perceptions on the basis of each robot reliability, but this was only partially accomplished since research units had the unconscious tendency to attribute a very low "reliability" to other robots with respect to their own, therefore invalidating the mechanisms (this should give you the feeling of how "heterogeneous" ART'99 was).

During an experiment, while debugging Bart with the graphical monitor, we observed the following behaviour, which was subsequently repeated and recorded: Bart is provided with a camera, but nobody has pressed the button that enables motors, and consequently the robot cannot move. Homer lies unused in a corner, its camera has been borrowed for some other robot, and somebody forgot its motors switched on. As the start message is published, Bart stands still, and Homer – to our surprise - tries to reach the ball in its place. Obviously, since Homer is not localized with respect to the same reference frame than Bart, Bart's perceptions are not sufficient to correctly reach the ball; we thus repeat the experiment after opportunely positioning Homer in the field (the experiment was evocatively baptized "the limp and the blind"). This time, Homer uses the vision messages sent by Bart to correctly locate the ball, it reaches it, and finally it pushes the ball towards the opponent goal and scores. In Fig. 6, we have emphasised the messages produced by the experts running in Homer and Bart during a 100ms interval in two different situations: Homer going to the ball and Homer carrying the ball. In Fig. 7 the experts in execution and the type of messages distributed are listed. A part of these messages is shared in broadcast (implicitly on the basis of the EIEP) on the local network (lines in bold in Fig. 6) and can therefore be received by the other robots or by a supervision workstation. Notice that only Bart's expert n. 4 (ETVision) produces vision messages (MSG_BALL_PERCEPTION). Such messages are received by Homer and, in particular, by the expert n. 3 (ETArbiter). In the absence of analogous locally produced messages,

ETArbiter uses them to activate different behaviours (experts ETGoToBall, ETCarryBall, ETNearWall etc.) depending on the situation in which the robot is in.

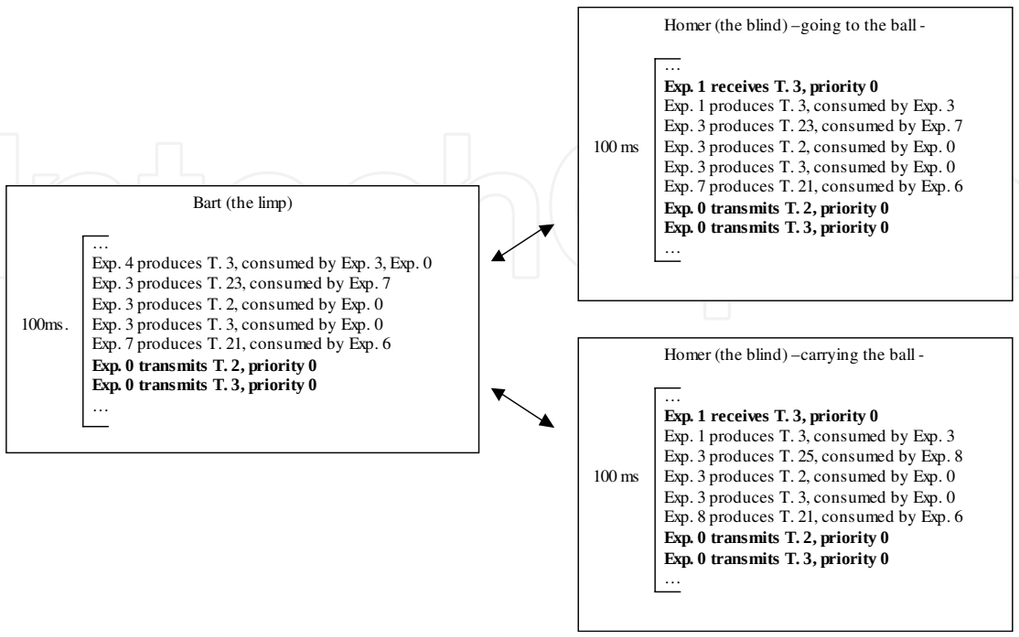

Fig. 6. "The limp and the blind": message exchanged between Bart and Homer.

Experimental data of the role-assignment algorithm have been collected by University of Rome during the European championship in year 2000; log files, compared with the a visual review of the game, show how roles are assigned to robots depending on the number of players currently available in the field. For example, when only two robots are working, the most important roles, i.e., "Striker" and "Defender", are correctly assigned, whereas the role "Supporting" is left unassigned. Fig. 8 on the left shows the percentage of time in which a role is assigned to at least one robot during different matches. Fig. 8 on the right shows the percentage of time in which at least one robot was present in the Forward, Middle and Backward zone. It can be noticed that, even if no explicit coverage algorithm has been implemented, yet the robot are reasonably distributed all over the field, with at least one robot almost constantly defending the goal in the Backward zone. As a result of the stabilizing mechanism, other analyses show that, on average, robots change role every 10 seconds. Some major problems come to the fact that the utility functions are subjectively evaluated, and it is possible that a robot overestimate its ability to play a role without being really suited to it. Consider a robot which incorrectly sees the ball very close, and therefore improperly takes the role "Striker"; the coordinated behaviour of the team turns out to be very sensitive to individual faults in hardware and algorithms. To obviate that, we exploit a mechanism similar to "acquiescence" described in (Parker, 1998); a metric is introduced to measure the individual success of each robot in playing its role, and a factor is consequently introduced in the utility function to take into account the progress made. Once again, the metric to be adopted is task dependent, and it is heuristically chosen through experimental validation. In case of a "Striker", we can take in account the perceived distance from the ball, which must decrease – on average - when the robot is approaching it, or the distance from the goal. Experiments show that this very simple mechanism allows reducing the effect of wrong role-assignments, thus improving the performance of the robot coordination algorithm.

System Experts

Number	Description	<*>	<^>
0	RxServer	P	100000
1	TxServer	P	100000

User Defined Experts

Number	Description	<*>	<^>
2	ETPioneer	P	100000
3	ETArbiter	P	100000
4	ETVision	P	100000
5	ETFindBall	SP	100000
6	ETMove	SP	100000
7	ETGoToBall	SP	100000
8	ETCarryBall	SP	100000
9	ETNearWall	SP	100000

* - Periodic (P), Sporadic (SP)
^ - Period or minimum inter-arrival time

Internal Messages

Description	Type Number
ActivateMove	21
ActivateKick	22
ActivateGoToBall	23
ActivateFindBall	24
ActivateCarryBall	25
ActivateNearWall	26

Internal and External Messages

Description	Type Number
MSG_STARTSTOP	1
MSG_POSITION	2
MSG_BALL_PERCEPTION	3
MSG_OPPONENTS_PERCEPTION	4
MSG_GOAL_PERCEPTION	5

Fig. 7. Left: active experts on Bart and Homer. Right: message types.

Match	Striker (%)	Supporting (%)	Defender (%)
1	82.9	39.5	98.2
2	84.6	98.0	80.8
3	87.6	38.5	90.0
4	89.5	81.8	96.9
5	93.5	84.1	98.9
Avg	87.6	68.3	93.0

Match	Forward (%)	Middle (%)	Backward (%)
1	78.8	68.7	97.8
2	65.4	68.8	98.6
3	53.2	54.5	99.3
4	23.5	80.4	93.1
5	64.1	72.1	98.9
Avg	57.0	68.9	97.5

Fig. 8. Left: Statistics for role assignment. Right: Statistics for each area coverage.

Finally, ETHNOS has been widely exploited in the ART team from the software engineering perspective, providing support for code reuse, component-based design, run-time debugging at a behavioural level, etc. Despite all the differences in robots hardware and software and the large number of people involved, expert de-coupling and object orientation have allowed to easily put together components developed by different people in the same group and, in a limited number of cases, across different groups. Whenever this happened, expert real-time scheduling and analysis allowed evaluating solutions in different architectural contexts, whereas inter-expert and inter-robot communication protocols allowed programmers to ignore details on how information is exchanged at a lower level. Even if we cannot provide a quantitative evaluation of ETHNOS "usability" in this sense, the importance of these properties was particularly evident in the one-week training phase before RoboCup.

5. Future works

In these years, ETHNOS properties have been exploited in many other robotic systems. Fig. 9 shows the robot Staffetta, a platform which has been especially designed for autonomous transportation within hospitals, but has been also adapted to different indoor scenarios and applications such as tour-guiding in museums and public expositions, autonomous mobile

surveillance, etc. By extending all the concepts that have been introduced so far, we now face all the problems related to autonomy in a fully distributed Multi Agent perspective. Robots are thought of as mobile physical agents within an intelligent environment (we metaphorically call it an "Artificial Ecosystem – AE") where they coexist and cooperate both with other robots and with several fixed physical agents, controlled by hundreds of micro-processors and connected through a building automation network. Fixed agents are intelligent sensing/actuating devices with different roles: they handle automated doors and elevators, provide an aid for autonomous navigation and localization, detect emergency situations such as fires, or liquid leak, etc.; in mobile surveillance applications, they handle passive infrared sensors, RFID readers to distinguish allowed personnel from intruders, cameras, etc.

Fig. 9. Left: Staffetta at Gaslini Hospital. Right: Staffetta at Villanova d'Albenga Airport.

There are both theoretical and practical justifications to this approach; if one consider existing mobile robotic systems in literature, only few of them have proven to be really autonomous in a generic, non structured environment, i.e. able to work continuously, for a long period of time, carrying out its tasks with no performance degradation or human intervention – the "six months between missteps" that (Moravec, 1999) advocates. Following (Brooks, 1990), an explanation can be given: the kind of intelligence we can expect from biological beings heavily relies on the sensors and actuators they are provided with, and the way in which they evolved in time in tight relation with the environment where they live. Given this assumption, there is no wonder that attempts fail, when trying to build an autonomous artificial being which is able to "live" in the same environments where biological beings live: the current technology in sensors and actuators is not adequate to such environments. Thus, we can choose between two different approaches: reproducing biological beings (which implies reproducing also their sensors and actuators) or building robots which have limited sensing and actuating abilities, but which operate within an environment which is well-suited for them. We claim that, whereas part of the current research aims at building very expensive and complex humanoid robots, it is possible – and, in the short time, preferable – to put robots back in their habitat, which allows to build simpler, cheaper, and more reliable autonomous systems. Ambient Intelligence and related disciplines are nowadays receiving a great attention by the scientific

community, and the idea of using cameras or other sensors to track robots in the environment has been proposed; however, the problem is rarely faced in an integrated perspective, by addressing – as we do – the architecture of a system composed of many robots and many fixed sensors distributed in the environment (recent exceptions are Broxvall et. al, 2006; NRF).

In the Artificial Ecosystem, both robots and intelligent devices are handled by ETHNOS experts, which communicate on a common message board through the publish/subscribe EIEP and are given a high level of autonomy; since autonomy and intelligence are not only a characteristic of robots, but are distributed throughout the environment (Fig. 10 on the left), we say that robots are autonomous but not "autarchic". Next, we extend the concept of intelligent devices to the sensors and actuators on board the robot, allowing to control the robot at two different levels (thus increasing fault-tolerance, Fig. 10 on the right): on the lower level a higher reactivity is achieved through on-board intelligent devices which control sensors and actuators; on a higher level, sensorial inputs are collected by experts running on the onboard computer and performing more sophisticate computations before issuing control to actuators.

For on-board and off-board devices we rely on fieldbus technology and, in particular, Echelon LonWorks, a standard for building automation: experts are executed on LonWorks Neuron Chips, microprocessors with a low computational power and a dedicated OS. ETHNOS publish/subscribe protocol is implemented in LonWorks through the concept of "network variables", i.e. strongly typified data which are propagated on the bus and delivered to all devices which have subscribed to that kind of information (the "network variables" abstraction is provided by the Application Layer of the LonWorks communication protocol). All the control nodes in the system (i.e. sensors/actuators both on board the robot and distributed in the building) are connected to the same fieldbus, by adopting a low bandwidth infrared channel for the communication between on-board devices and the building network.

The architectural infrastructure of the system has been extensively tested, and some case studies are available for evaluation; our past set-up at the Gaslini Hospital in Genova (years 2000–2002), the mobile surveillance system ANSER at the Villanova d'Albenga airport, and a permanent set-up in our Department at University of Genova (Fig. 9). In these scenarios, we mostly implemented devices for self-localization ("intelligent beacons") and for the control of automated doors and elevators ("intelligent gatekeepers"). However, to fully take benefit of the approach and draw conclusions about its properties, more work has to be done. Visionary scenarios include: advanced coordinated behaviours, involving both on-board and off-board intelligent devices; multipurpose intelligent devices, able to assume different roles and perform different tasks according to a given allocation strategy; a parasite robot which relies on the sensors of other robots, possibly without them being aware of that; a robot which is "possessed" by another robot or by an intelligent device, which takes control of its actuators and forces it to execute a given task; robots and intelligent devices which ask (or force…) humans to help them to execute a given task; etc. This "ubiquitous distribution" of competencies and capabilities in the environment, possibly with humans being involved, has not been investigated yet; the Artificial Ecosystem, however, is totally compatible and open to this new paradigm, thus definitely encouraging to explore it.

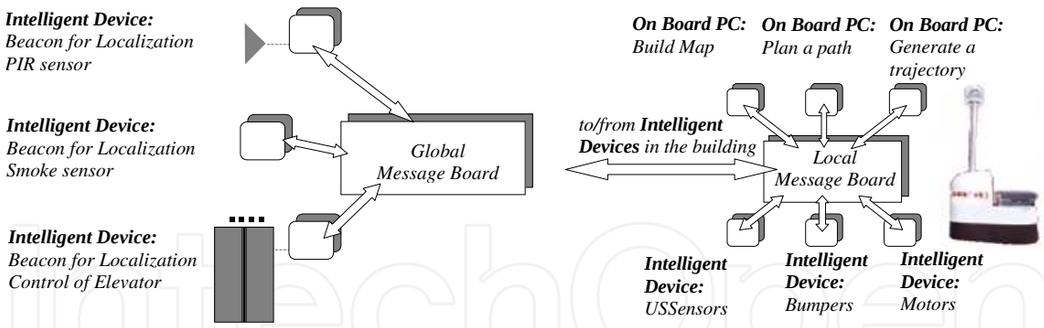

Fig. 10. Left: Intelligent devices in the building; Right: Intelligent devices on the robot.

6. Conclusions

We presented ETHNOS, a software architecture and a programming environment for the real-time control of distributed Multi Robot systems; its properties have been described as well as its influence on the development of a soccer team for RoboCup. Throughout the chapter, we have argued that the ART team in Stockholm 1999 has been an unique experience because of its extreme heterogeneity: players in the team differed both in the hardware and in the software, and were designed in total autonomy by different research units with almost no cooperation; however, after only one week of training before RoboCup, they were able to coordinate themselves on the soccer field and to win the second prize. ETHNOS contribution to achieving this is discussed.

In particular, to meet the expectations of all the research group involved, with many different point of view about robotic intelligence, ETHNOS does not imposes a specific cognitive architecture. In fact, it rather specifies the internal organization of software processes and their interactions, providing a framework to decompose complex Multi Robot systems into simpler software components. For this reasons, it is also referred to as an "Operating System" for robotics. All the design choices made have been discussed in details, providing both theoretical justifications and experimental validations. The system's versatility is outlined as well, by showing how the traditional Multi-Robot paradigm can be extended to include a broader conception of Multi-Robot and Multi-Agent coordination: the Artificial Ecosystem approach, which takes inspiration from research in Ambient Intelligence to propose a new paradigm for the design of efficient, reliable, and autonomous (but not autarchic) mobile robotic systems.

7. Acknowledgements

I wish to thank Prof. Zaccaria and Dr. Piaggio, with whom I designed ETHNOS; Prof. Nardi, coordinator of ART in 1999 and 2000, Prof. Adorni, Prof. Bonarini, Prof. Chella, Prof. Pagello, Prof. Iocchi and all the students of the Universities of Rome, Parma, Milano, Palermo, and Padova for their significant contribution in the experimentation of ETHNOS, and for some of the data and the photos shown here.

8. References

Acar, E.U.; Choset, H.; Zhang, Y. & Schervish, M.S. (2003). Path Planning for Robotic Demining: Robust Sensor-based Coverage of Unstructured Environments and Probabilistic Methods. *Int. Journal of Robotic Research*, Vol. 22, No. 7-8, 441-466

Alami, R.; Chatila, R.; Fleury, S.; Herrb, M.; Lngrand, F.; Khatib, M.; Morisset, B.; Moutarlier, P. & Simeon, T. Around the lab in 40 days, *Proc. of the IEEE Int. Conference on Robotics Automation*, pp. 88–94, April 2000, San Francisco, CA.

Arkin, R.C. (1990). Integrating behavioral, perceptual, and world knowledge in reactive navigation. *Robotics and Autonomous Systems*, Vol. 6, No. 1-2, 105–122.

Arkin, R.C. (1992). Cooperation without communication: Multiagent schema-based robot navigation. Journal of Robotic Systems, Vol. 9, No. 3, 351–364.

Bahadori, S.; Calisi, D.; Censi, A.; Farinelli, A.; Grisetti, G.; Iocchi, L. & Nardi, D. (2005). Autonomous systems for search and rescue. *Rescue Robotics, A Birk, S. Carpin, D. Nardi, Jacoff A., and S. Tadokoro (Eds.)*, Springer-Verlag, 2005.

Balch, T. & Arkin, R.C. (1994). Communication in Reactive Multiagent Robotic Systems, *Autonomous Robots*, Vol. 1, No. 1, 27-52.

Balch, T. & Arkin, R.C. (1998). Behavior-based formation control for multirobot teams. *IEEE Transactions on Robotics and Automation*, Vol. 14, No. 6, 926-939.

Balch, T. (2000). Hierarchic Social Entropy: An Information Theoretic Measure of Robot Group Diversity, *Autonomous Robots*, Vol. 8, No. 3, 209–237.

Balch, T. & Parker, L.E. (2002). *Robot Teams: From Diversity to Polymorphism, T. Balch and L.E. Parker (Eds.)*, A K Peters Ltd.

Barfoot, T. & Clark, C. (2004). Motion Planning for Formations of Mobile Robots. *Journal of Robotics and Autonomous Systems*, Vol. 46, No. 2, 65-78.

Beard, R. W.; Lawton, J. & Hadaegh, F.Y. (2001). A feedback architecture for formation control. *IEEE Transactions on Control System Technology.*, Vol. 9, 777–790.

Berman, S.; Edan, Y. & Jamshidi, M. (2003). Navigation of decentralized autonomous automatic guided vehicles in material handling. *IEEE Transactions on Robotics and Automation*, Vol. 19, No. 4, 743- 749.

Bonasso, R.; Firby, R.; Gat, E.; Kortenkamp, D.; Miller, D. & Slack, M. (1997). Experiences with and architecture for intelligent, reactive agents. *Journal of Experimental and Theoretical Artificial Intelligence*, Vol. 9, No. 2, 237-256.

Brooks, R.A. (1986). A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, Vol. 2, No. 1, 14-23.

Brooks, R.A. (1990). Elephants Don't Play Chess. *Journal of Robotics and Autonomous Systems*, Vol. 6, No. 1-2.

Broten, G.; Monckton, S.; Giesbrecht, J. & Collier, J. (2006). Software Systems for Robotics An Applied Research Perspective. *International Journal of Advanced Robotic Systems*, Vol. 3, No. 1, 11-16.

Broxvall, M.; Gritti, M.; Saffiotti, A.; Seo, B.S. & Cho, Y.J. PEIS ecology: Integrating robots into smart environments, *Proc. of the 2006 Int. Conf. on Robotics and Automation*, pp 212- 218, May 2006, Orlando, FL.

Brumitt, B.; Stentz, A.; Herbert, M., & the CMU UGV Group (2001). Autonomous Driving with Concurrent Goals and Multiple Vehicles: Mission Planning and Architecture, *Autonomous Robots*, Vol. 11, No. 2, 103–115.

Burgard, W.; Moors, M.; Stachniss, C. & Schneider, F. (2005). Coordinated multi-robot exploration. *IEEE Transactions on Robotics*, Vol. 1, No 3, 376-385.

Buttazzo, G. C. *Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications*. Kluwer Academic Publishers, 1997.

Choset, H. (2001). Coverage for robotics—A survey on recent results. *Annals of Mathematics and Artificial Intelligence*, Vol. 31, 113–126.

Côté, C.; Brosseau, Y.; Létourneau, D.; Raïevsky, C. & Michaud, F. (2006). Robotic Software Integration Using MARIE. *Int. Journal of Advanced Robotic Systems*, Vol. 3, No. 1, 55-60

De Pascalis, P; Ferraresso, M; Lorenzetti, M; Modolo, A; Peluso, M; Polesel, R; Rosati, R.; Scattolin, N.; Speranzon, A. & Zanette, W. (2001). Golem Team in Middle-Sized Robots League, *RoboCup 2000: Robot Soccer World Cup IV*, *Lecture Notes In Computer Science, Vol. 2019*, Springer-Verlag, London.

Drgoul, A. & Ferber, J. (1993). From tom thumb to the dockers: Some experiments with foraging robots. *From Animals to Animats 2: Proceedings of the 2nd Int. Conf. on Simulation of Adaptive Behavior*, pp 451-459.

Fontan, M.S. & Matarić, M.J. (1996). A study of territoriality: The role of critical mass in adaptive task division. *From Animals to Animats 4: Proceedings of the 4th Int. Conf. on Simulation of Adaptive Behavior,* MIT Press, Cambridge, MA.

Fox., D; Burgard, W.; Kruppa., H. & Thrun., S. (2000). A Probabilistic Approach to Collaborative Multi-Robot Localization. *Autonomous Robots*, Vol. 8, No. 3, 325–344.

Gerkey, B. & Mataric, M. (2002). Sold! auction methods for multirobot coordination. *IEEE Transactions on Robotics and Automation*, Vol. 18, No. 5, 758–768.

Gerkey, B.; Vaughan, R. T. & Howard, A. (2003). The player/stage project: Tools for multi-robot and distributed sensor systems. *Proc. of the 11th Int. Conf. on Advanced Robotics*, pp. 317-323.

Gustafson S. & Gustafson, D.A. (2006). Issues in the scaling of multi-robot systems for general problem Solving, *Autonomous Robots*, Vol. 20, No. 2, 125–136.

Iocchi, L; Nardi, D.; Piaggio, M. & Sgorbissa, A. (2003). Distributed Coordination in Heterogeneous Multi-Robot Systems. *Autonomous Robots*, Vol. 15, No. 2, 155-168.

Jeffay, K.; Stanat, D.F. & Martel, C.U. (1991) On Non-preemptive Scheduling of Periodic and Sporadic Tasks, *IEEE Real time Systems Symp.*, pp 121-139, December 1991.

Jung, D. & Zelinski, A. (2000). Grounded Symbolic Communication between Heterogeneous Cooperating Robots. *Autonomous Robots*, Vol. 8, No. 3, 269–292.

Kitano, H.; Asada, M.; Kuniyoshi, Y.; Noda, I.; Osawa, E. & Matsubara, H. (1997). Robocup: A challenging problem for AI and robotics, in *RoboCup-97: Robot Soccer World Cup I. Lecture Notes in Artificial Intelligence, Vol. 1395.* Springer-Verlag, New York, 1998, 1–19.

Konolige, K. & Myers, K. (1998). The saphira architecture for autonomous mobile robots. *Artificial Intelligence and Mobile Robots: Case Studies of Successful Robot Systems, D. Kortenkamp, RP Bonasso, and R. Murphy (Eds.)*, MIT Press, March 1998.

Kurazume, R., & Hirose, S. (2000). Development of a Cleaning Robot System with Cooperative Positioning System, *Autonomous Robots*, Vol. 9, No. 3, 237–246.

Lehoczky, J. P.; L. Sha; & J. K. Strosnider. (1992). Enhanced Aperiodic Responsiveness in Hard RealTime Environments. *Proceedings of the Real Time Systems Symposium,* pp. 110-123, , December 1992, Phoenix, AZ.

Madhavan R.; Fregene K. & Parker, L.E. (2004). Distributed Cooperative Outdoor Multirobot Localization and Mapping. *Autonomous Robots*, Vol. 17, No. 1, 23–39.

Matarić, M.J.; Nilsson, M. & Simsarian, K.T. (1995). Cooperative multi-robot box-pushing, *Proc. of the Int. Conf. on Intelligent Robots and Systems*, pp. 556–561, May 1995, Pittsburgh, PA.

Moravec, H., Rise of the Robots, *Scientific American*, December 1999, pp. 124-135

Musliner, D.; Durfee, E. & Shin, K. (1993). Circa: A cooperative, intelligent, real-time control architecture. *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. 23, No. 6.

Nesnas, I. A.D.; Simmons, R.; Gaines. D.; Kunz., C.; Diaz-Calderon, A.; Estlin, T.; Madison, R.; Guineau, J.; McHenry, M.; Shu, I-H. & Apfelbaum, D. (2006) CLARAty: Challenges and Steps Toward Reusable Robotic Software, *International Journal of Advanced Robotic Systems*, Vol. 3, No. 1, 23-30.

NRF, Network Robot Forum, www.scat.or.jp/nrf/English/.

Parker, L.E. (1998). Alliance: An architecture for fault tolerant multi-robot cooperation. *IEEE Transactions on Robotics and Automation*, Vol. 14, No. 2, 220–240.

Parker, L.E. (2000). Current state of the art in distributed robot systems. *Distributed Autononomous Systems 4, L.E. Parker, G. Bekey and J. Barhen (Eds.)*, Springer-Verlag.

Parker, L.E. (2002). Distributed Algorithms for Multi-Robot Observation of Multiple Moving Targets. *Autonomous Robots*, Vol. 12, No. 3, 231–255.

Pirjanian, P.; Huntsberger, T.; Trebi-Ollennu, A.; Aghazarian, H.; Das, H.; Joshi, S. & Schenker, P. (2000). Campout: a control architecture for multi-robot planetary outposts. *Proc. of the SPIE Symposium on Sensor Fusion and Decentralized Control in Robotic Systems III*, pp. 221-230, Boston, MA.

Rekleitis, I.M.; Dudek, G. & Milios, E.E. (2000). Graph-based exploration using multiple robots. *Distributed Autonomous Robotics Systems 4, L.E. Parker, G.W. Bekey, and J. Barhen (Eds).*, Springer-Verlag, Berlin.

Rybski, P. E.; Stoeter, S. A.; Gini, M.; Hougen, D. F.; & Papanikolopoulos, N. (2002). Performance of a distributed robotic system using shared communications channels. *IEEE Trans. on Robotics and Automation*, Vol. 22, No. 5, 713--727.

Schlegel, C. (2006). Communication Patterns as Key Towards Component-Based Robotics, *International Journal of Advanced Robotic Systems*, Vol. 3, No. 1, 49-54.

Schmitt, T.; Hanek, R.; Beetz, M.; Buck, S. & Radig. B. (2002). Cooperative probabilistic state estimation for vision-based autonomous mobile robots. *IEEE Transactions on Robotics and Automation*, Vol. 18, No. 5, 670-684.

Schneider, F.E.; Wildermuth D. & Kräußling, A. (2005). Discussion of Exemplary Metrics for Multi-Robot Systems for Formation Navigation. *International Journal of Advanced Robotic Systems*, Vol.2, No. 4, 345-353.

Sgorbissa, A. & Arkin, R.C. (2003). Local Navigation Strategies for a Team of Robots, *Robotica*, Vol. 21, No. 5, 461-473.

Siegwart, R.; Arras, K.O.; Bouabdallah, S.; Burnier, D.; Froidevaux, G.; Greppin, X.; Jensen, B.; Lorotte, A.; Mayor, L.; Meisser, M.; Philippsen, R.; Piguet, R.; Ramel, G.; Terrien, G.; & Tomatis, N. (2003). Robox at Expo.02: A LargeScale Installation of Personal Robots. *Robotics and Autonomous Systems*, Vol. 42, No. 3-4, 203-222.

Simmons, R.; Smith, T.; Dias, M.B.; Goldberg, D.; Hershberger, D.; Zlot, R. & Stentz, A. (2002) A layered architecture for coordination of mobile robots. *Multi-Robot Systems: From Swarms to Intelligent Automata, A. Schultz and L. Parker (Eds.)*, Kluwer Academic Publisher, May 2002.

Stone, P. & Veloso, M. (1999). Task decomposition, dynamic role assignment, and low-bandwidth communication for real-time strategic teamwork. *Artificial Intelligence*, Vol. 110, No. 2, 241–273.

Utz, H.; Sablatnög, S.; Enderle, S. & Kraetzschmar, G. (2002). Miro—Middleware for Mobile Robot Applications, *IEEE Transactions on Robotics and Automation*, Vol. 18, No. 4, 493-497.

Vidal, R.; Shakernia, O.; Jin, H.; Hyunchul, D.; & Sastry, S. (2002). Probabilistic Pursuit-Evasion Games: Theory, Implementation, and Experimental Evaluation. *IEEE Trans. Robotics and Automation*, Vol. 18, No. 5, 662- 669.

Weigel, T.; Gutmann, J.-S.; Dietl, M.; Kleiner, A. & Nebel. B. (2002). CS freiburg: Coordinating robots for successful soccer playing. *IEEE Trans. on Robotics and Automation*, Vol. 18, No. 5, 685-699.

Werger, B.B. & Matarić, M.J. (2000). Broadcast of local eligibility for multitarget observation, *Distributed Autonomous Robotic Systems 4, L. E. Parker, G. Bekey, and J. Barhen (Eds.)*, Springer-Verlag.

Yamashita, A.; Arai, T.; Ota, J. & Asama, H. (2003). Motion planning of multiple mobile robots for Cooperative manipulation and transportation. *IEEE Transactions on Robotics and Automation*, Vol. 19, No. 2, 223-237.

Mobile Robotics, Moving Intelligence

Edited by Jonas Buchli

ISBN 3-86611-284-X

Hard cover, 586 pages

Publisher Pro Literatur Verlag, Germany / ARS, Austria

Published online 01, December, 2006

Published in print edition December, 2006

This book covers many aspects of the exciting research in mobile robotics. It deals with different aspects of the control problem, especially also under uncertainty and faults. Mechanical design issues are discussed along with new sensor and actuator concepts. Games like soccer are a good example which comprise many of the aforementioned challenges in a single comprehensive and in the same time entertaining framework. Thus, the book comprises contributions dealing with aspects of the Robotcup competition. The reader will get a feel how the problems cover virtually all engineering disciplines ranging from theoretical research to very application specific work. In addition interesting problems for physics and mathematics arises out of such research. We hope this book will be an inspiring source of knowledge and ideas, stimulating further research in this exciting field. The promises and possible benefits of such efforts are manifold, they range from new transportation systems, intelligent cars to flexible assistants in factories and construction sites, over service robot which assist and support us in daily live, all the way to the possibility for efficient help for impaired and advances in prosthetics.

How to reference

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

Antonio Sgorbissa (2006). Multi-Robot Systems and Distributed Intelligence: The ETHNOS Approach to Heterogeneity, Mobile Robotics, Moving Intelligence, Jonas Buchli (Ed.), ISBN: 3-86611-284-X, InTech, Available from: http://www.intechopen.com/books/mobile_robotics_moving_intelligence/multi-robot_systems_and_distributed_intelligence__the_ethnos_approach_to_heterogeneity

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