Agents for Cooperative Heterogeneous Mobile Robotics: a Case Study*

Patricio Nebot
Robotic Intelligence Lab
Jaume-I University
al068259@uji.es

Daniel Gómez
Robotic Intelligence Lab
Jaume-I University
al003995@uji.es

Enric Cervera
Robotic Intelligence Lab
Jaume-I University
ecervera@uji.es

Abstract – *In the last years, there is a greater interest in systems of multiple autonomous robots for the accomplishment of cooperative tasks. This fact has motivated the implementation of a distributed architecture for a team of heterogeneous mobile robots. Such architecture must have the capacity to allow to the team of robots the accomplishment of cooperative tasks; the proposed solution consist of a multiagent system for the implementation of this distributed architecture. Also, two practical applications of cooperative tasks with the created architecture are sketched: cooperative navigation and localization and cooperative visual servoing.*

Keywords: Mobile robots, heterogeneity, agents, cooperation.

1 Introduction

The advances in mobile robotics, computing power and wireless communications have turned feasible the development of communities of autonomous robots but with capacity to make cooperative tasks.

Teamworking is an essential capability for multiple mobile robots[7]. Having one single robot with multiple capabilities (vision, laser, manipulator, computer) may be a waste of resources. Different robots, each one with its own configuration, are more flexible, robust and cost-effective. Moreover, the tasks to make may be result too complex for one single robot, and having multiple robots it is possible to increase the effectiveness.

To establish mechanisms of cooperation between robots implies to consider a problem of design of cooperative behavior: given a group of robots, an environment and a task, how the cooperation must be carried out. Such problem implies several challenges, emphasizing among them the definition of the architecture of the group. The multiagent systems are the natural environment for such groups of robots, making possible the fast implementation of powerful architectures for the specification and execution of tasks.

This article describes the implementation and development of a distributed architecture for the programming and control of a team of coordinated heterogeneous mobile robots, which are able to collaborate among them and with people in the accomplishment of tasks of services in daily environments.

The most important characteristic of the project is the *cooperation* between the diverse robots that forms the team to make tasks of coordinated way. Our work represents a qualitative leap with the introduction of a cooperation architecture between different robots heterogeneous.

The rest of the article it is structured in the following form: in Section 2 a review of works related to the present project will be made; in Section 3 the technical aspects of the robots, specifying their characteristics and the differences among them that produce the heterogeneity, will be approached; in Section 4 the advantages of use agents in front of objects will be seen, and next the architecture created for the project will be described; in Section 5 the experimental results obtained from the use of the created architecture will be emphasized; and finally in Section 6 it will be enumerated the conclusions of the present work.

2 State of the art

There are several works where are shown architectures and systems for the control, management and programming of teams of robots for the accomplishment of cooperative tasks. Among them it is possible to emphasize the following ones:

ALLIANCE is an architecture oriented to the cooperation of a small-medium team of heterogeneous robots, with little communication among them[9]. It assumes that robots are relatively able to discern the effects of their actions and those of the rest of agents, either through its perception or through communication. Individual robots act based on behaviors or sets of behaviors to make their tasks.

The project *MICRobES* is an experiment of collective robotics that tries to study the long time adaptation of a micro-society of autonomous robots in an environment with humans. Robots must survive in this environments as well as cohabit with his people[10].

The group of intelligent robotics and computer vision of the Rovira i Virgili University has created an architecture for dynamic physical agents, *DPA (Dynamic Physical Agents)*[8]. A physical agent is the result of integrate a software agent in an autonomous hardware. This hardware

*0-7803-7952-7/03/$17.00 © 2003 IEEE.

is frequently a mobile robot. DPA architecture shows the agent like a modular entity with three levels of abstraction: control module, supervisor module and agent module. The architecture's operation is based on two basic processes for each module: the process of introspection and the one of control, and in a protocol of intermodular dialogue which allows the exchange of information between modules.

The Robotic Systems Lab of the Australian National University has developed an architecture for mobile robots, *ABBA (Architecture for Behaviour Based Agents)*[5]. The purpose of the project is to design an architecture to model the robot's behavior so that it can select reactive behaviors of low level for his survival, to plan high level objectives oriented to sequences of behaviors, to carry out spatial and topological navigation, and to plan cooperative behaviors with other agents.

The previous works implement architectures and systems so that teams of mobile robots can make cooperative tasks. Our work consists of the implementation of an architecture of such type, adding the versatility and power of the multiagent systems together with distributed systems, for the resolution of cooperative tasks for a group of heterogeneous robots

3 Technical aspects of the robots

For the development of the project, we use a team consisting of 6 robots Pioneer-2 of medium size and a Powerbot robot of greater size, all of them made by ActivMedia Robotics[1].

Figure 1: Team of robots used in the project.

Among the robots exist different characteristics, constituting therefore a heterogeneous group. In this group, they vary the dimensions of robots and the different accessories added to robots (cameras, laser, grippers), moreover, ones incorporate an Intel Pentium III processor and others have solely a microcontroller.

All robots of the team maintain the same logical design based on a microcontroller which transfers the readings of the standard elements of the robot (ultrasound sensors, wheel encoders) via RS232, by cable or radio, to a computer client where the applications are executed. This computer can be located on board of the robot or not, in which case the communication is made by radio-modem.

In robots with on board computer, the computers are connected by means of 802.11b cards to each other, as well as with the base stations that access to the local network of the university, and to Internet.

Next, the technical characteristics of robots are detailed[1]:

- *1 Powerbot*: autonomous mobile robot of high capacity. It has on board computer with connection to the wireless network. In addition it has 28 ultrasound sensors, 14 forwards and 14 backwards, it has front and back contact sensors, and frontal infrared sensors to detection of stairs or holes in the ground.
- *2 Pioneer-2 provided with vision*: autonomous mobile robots which have on board computer with connection to the wireless network, and system of video capture through a PTZ camera. Also they have 8 ultrasounds sensors on front and a gripper of two degrees of freedom.
- *1 Pioneer-2 provided with laser scanner*: autonomous mobile robot that has on board computer with connection to the wireless network, and has a planar scanner by SICK laser. Moreover it has 8 front and 8 back ultrasound sensors, and one gripper of two degrees of freedom.
- *3 Pioneer-2*: semi-autonomous mobile robots, because of they have not on board computer, reason why its microcontroller must be in permanent RS232 communication (by cable or radio) with an external computer for its control. They have 8 frontal ultrasound sensors and a gripper of 2 degrees of freedom.

4 Implementation of the architecture

As mentioned previously, the objective of our work is the development of a distributed architecture for the programming and control of a group of multiple heterogeneous mobile robots able to cooperate among them and with people for the accomplishment of tasks of services in daily environments.

Pioneer mobile robots, as well as the Powerbot robot, already has a very complete programming architecture with libraries for all their accessories. This architecture is made up by two layers, the inferior layer is formed by the ARIA API[1], which is in charge of the management of the requests from programs to the components that form the robot; and the superior layer is form by a control system called Saphira[1, 6], that provides services of a superior level as can be the location or navigation of the robot. Nevertheless, this programming architecture is oriented to the local or remote process from an only controller, without defining mechanisms of mutual collaboration.

The task to do will be to add a layer to that architecture to share of simple and effective way the resources of each robot among all the group. That is to develop an intermediate level (middleware) between the functions of the robot (ARIA + Saphira) and the applications. This can be seen in Figure 2.

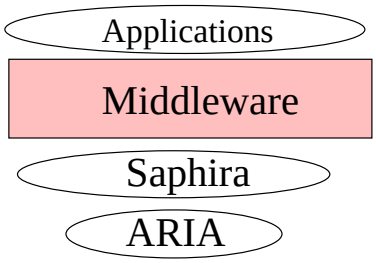

Figure 2: General diagram of the architecture.

For the resolution of the described task, it is considered an approach oriented to agents, by means of the use of an agent development tool[12].

This approach supposes that the additional layer must be written in Java, owing to the tool used is written in Java. Because of the inferior layer of our architecture (ARIA + Saphira) is written in C/C++, it's necessary to use a tool that allows to handle this heterogeneity of languages. For such aim, JNI (Java Native Invocation) is used. It allows to incorporate fragments of code written in C/C++ in our Java programs. Thus, it will be possible to use the compiled code of the base layer (ARIA + Saphira) on the layer to implement.

4.1 Agents architecture

This approach implies the implementation of the middleware layer using a system multiagent (MAS). MAS are systems composed of multiple interacting computational elements, called agents. Agents are computational systems with two important capacities: they must be able to make autonomous actions (to decide by themselves that they need to make to satisfy their objectives), and they must be able to interact with other agents by means of some type of cooperation, collaboration, negotiation... [13]

This option has been adopted due to the advantages that offer the agents in front of the objects. These advantages cause that the agents are more appropriate for the implementation of concurrent and distributed systems[2]. Some of these advantages are:

- The objects have data and behaviors, whereas the agents have in addition beliefs, desires and intentions.
- The objects execute methods sequentially, whereas the agents reason about their beliefs in order to select the plan which could manage to satisfy its objectives.
- The objects simply responds to messages and events, whereas the agents also are able to act directed by their own objectives without the necessity of external incentives.
- The objects do not have control over their behavior, whereas the agents are flexible, that is, they can change their behavior in execution time to solve unexpected situations.
- The agents coordinate their actions with other agents through conversations to obtain individual and global objectives.

In summary, the agents can be seen like autonomous and intelligent objects, that are equipped with knowledge and reasoning capacities to satisfy several objectives.

All these advantages cause that it has been chosen to follow an approach of agents instead of an approach of objects. Therefore, the construction of the middleware layer of the architecture previously commented will follow a pattern of multiagent system. The advantages that a multiagent system offers for our project are:

- There is not a global system of control, since it is pretended to create a distributed system, in which each robot can accede to the capacities of another one.
- The data are decentralized, as each robot maintains its own data, but it is wanted that these data are accessible by all the team of robots.
- The computation is asynchronous, because of each robot decides to execute its own actions when it considers opportune.
- The agents dynamically decide what tasks to make and who makes them, by means of some sort of coordination or cooperation between the team of robots.

For the implementation of the multiagent system which is in charge of the middleware layer it has used a programming tool of multiagent systems called MadKit (Multi-Agent Development Kit)[4]. MadKit is a platform for development and execution of multiagent systems. It presents a mechanism for the construction of agents with communication capacities supporting KQML. Agents are distributed and have different attributes: perception, reactivity and knowledge.

MadKit is based on an organizational model called Aalaadin. As well, Aalaadin is based on three concepts: *agent*, *group* and *role*[3].

In this model, an agent is an active organization of communication that plays roles within the groups. This model does not put restrictions to the internal architecture of the agents. The designer of the agent is the responsible to choose the model of agent more appropriate. A group is a set of agents with similar functionality. Each agent can belong to one or more groups. A role is an abstract representation of the function or service of the agent. Each agent can have several roles, and each role is own of a group[3].

Moreover, MadKit implements another important concept, the *community*. A community is a group of interconnected MadKit kernels. Each kernel acts like node of a distributed environment. Thus, all the groups that are created must belong to one community[3].

In order to be able to implement in MadKit the middleware layer of our architecture, it has been created a community, the community "Seven Dwarfs", that includes all the groups of agents that they are formed. Inside this community there are so many groups as robots in the team, that is, exists one group "Doc", another group "Grumpy", another group "Sneezy", and thus up to seven groups. This can be seen clearly in Figure 3.

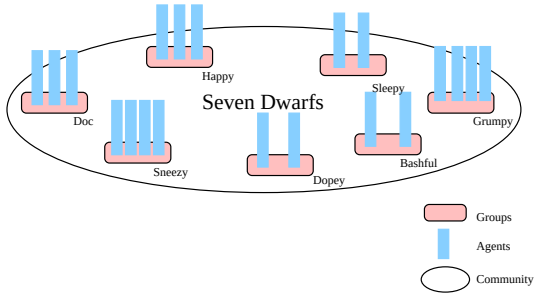

Figure 3: Structure of the "Seven Dwarfs" community.

Inside each of the groups that represent to the robots, there are the own agents of each robot. Inside the group, by means of the agents, an architecture of two layers has been created. In a first layer there are some agents who can be called "physical agents", since they are the responsibles to manage all the service requests that the rest of agents make to the physical robot itself. In this layer there are three agents, an agent who is in charge to manage all requests related to the movement of the robot, readings of the sonar, handling of the gripper..., that is, all that related to the robot base; an agent who is responsible to manage all that related to the camera, in those robots that have camera; and an agent in charge of all related to the laser, in those robots that have laser.

These "physical" agents are those that communicate with the base layer (ARIA + Saphira) of the global architecture by means of JNI. Each one of these three agents communicates with a specific device by means of the serial port to which that device is connected in the on board computer of the robot. Thus, the agent of the base communicates with the microcontroller of the robot by the serial port 1, the agent of the camera communicates with the camera by the serial port 2, and the agent of the laser communicates with the laser by the serial port 3.

Over the "physical" layer is constructed another layer constituted by several agents which accede to the "physical" agents of the previous layer. Each one of the agents of this layer is specialized in offering the services of a certain component of the robot. Thus, with the "base" physical agent communicate three agents, the *base* agent that is in charge to manage the movement of the robot and the state of the same one; the *sonar* agent that is responsible to manage the actions related to the sonar; and the *gripper* agent that is in charge of the actions of gripper handling. With the "camera" physical agent communicates two agents, the *camera* agent that is responsible of the physical movement of the camera; and the *vision* agent that is in charge to offer the minimum services of computer vision, as it can be to capture an image. Finally, with the "laser" physical agent communicates three agents, the *laser* agent that is responsible of the physical aspect of the laser, as it is to turn on it and to turn off it; the *location* agent that is in charge to offer the necessary services so that the robot can be located using the laser; and the *navigation* agent, that offers services so that the robot can navigate by environments also using the laser.

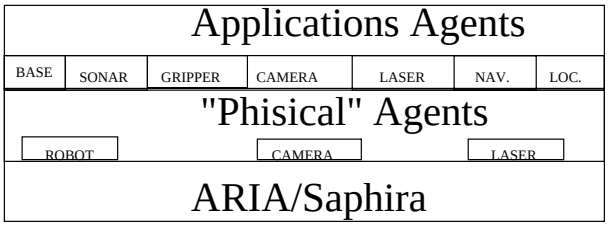

Figure 4: Structure of a group/robot.

As can be seen in the figure 4, over the two previous layers already is the layer corresponding to the applications, that can be implemented by means of agents. The applications, to be able to accede to the components of the robot, must accede to the agents of the superior layer of middleware, and these, as already it has been seen, are the responsible to communicate with the inferior layer which communicates directly with the robot.

5 Applications

Next, two applications that can use the described architecture are presented. First of them is one of the typical applications of any mobile robot, navigation and location, but extended to be made by a team of robots in cooperative way. Whereas the second application consists of a cooperative "visual servoing" application. To comment that we are even implementing the first prototypes, but we hope to have interesting results in very short time.

5.1 Cooperative navigation and location

Due to the changing conditions of the environment of work of the robots, to the technology of the used sensors and to the fact that each robot has a different equipment, it is possible that making a task, a robot momentarily loses its position with respect to the world that surrounds him.

In order to solve this problem, it is the cooperative location, where robots by means of visual references and the use of laser are able to return to be located, that is, to know its position with respect to a given environment. This is managed by means of a simple system, if the "lost" robot

has camera will try to see and to identify another robot, and by means of a simple calculus it will consider the position of this robot with respect to him. Next, it will communicate with that robot to know his position in the real world and, by means of simple calculus, it is able to establish his own position.

This also works in inverse mode, when the robot does not have camera then other that is equipped of it will look for him and it will help to find his position again, to be located.

Another situation is the cooperative navigation, in this case, a robot equipped with laser will go "opening way" providing to the others robots the map of the environment according to it is discovered. It is important to emphasize that here also is in operation the cooperative location, so that the others robots can know their position in this new environment

5.2 Cooperative visual servoing

The cooperative visual servoing consists in the control of the three mobile robots being based on visual information of a PTZ camera. One of the three robots takes a camera, and it gives the necessary information to a second robot to say it how to follow a third robot.

The purpose is that a robot provided with a laser makes a certain trajectory by an environment by means of navigation techniques. Behind this robot goes another one without no type of external sensor that allows it the navigation or following of the front robot. Next, a third robot provided with a camera follows them. This camera allows the robot to recognize the other two robots in the environment and to indicate to the robot of ahead as it must follow the first one, furthermore indicating itself like following both previous robots. That can be seen in Figure 5

Figure 5: Experiment of cooperative visual servoing.

6 Conclusions

In the present article we have described a distributed architecture for a team of heterogeneous mobile robots. This architecture joins the advantages of distributed computation and multiagent systems, thus it is appropriate for the implementation and execution of tasks that require certain collaboration by some robots of the team.

In addition, since the robot itself does not conform an agent, but each component of a robot is a specific agent, then the own components of a robot can be shared by all the group, with the consequent increase of the capacity of features of robots.

It is important to mention that a possible future application for the team of robots, implemented using the created architecture, could be the monitoring of buildings or factories. Robots could patrol the building detecting anomalies that could cause alarms.

Finally, another application in which already is working the laboratory is the librarian robot. Adding to the big robot an articulated arm, it acquires the capacity to manipulate books. Thus, this robot would be in charge in a library to look for and to extract of the book shelves the requested books, and helped by the other robots to be able to carry them to the users.

Acknowledgements

This paper describes research carried out at the Robotic Intelligence Laboratory of Universitat Jaume I. Support for this research is provided in part by the Ministerio de Ciencia y Tecnología under projects DPI2001-3801, HF2001-0112, by the Generalitat Valenciana under projects inf01-27, GV01-244, CTIDIA/2002/195, and by the Fundació Caixa-Castelló under project P1-1B2001-28.

References

[1] ActivMedia Robotics Software, Documentation and Technical Support, `http://robots.activmedia.com`

[2] Chi, J., Nie, L., Shinouda, M., *Actor Languages and Component-Based Design*, CS 854 Component-Based Software Design, Apendix III, 2002.

[3] Ferber, J., Gutknecht, O., *A Meta-model for the Analisys and Design of Organizations in Multi-agent Systems*, Third International Conference on Multi-agent Systems(ICMAS'98) Proceedings, pp. 128-135, IEEE Computer Society, 1998.

[4] Gutknecht, O., Ferber, J., MadKit Reference page, `http://www.madkit.org`

[5] Jung, D., Zelinsky, A., *An Architecture for Distributed Cooperative Planning in a Behaviour-Based Multi-Robot System*, Journal of Robots and Autonomous Systems,99 , Vol 26, No.2-3, pp. 149-174.

[6] Konolige, K., Myers, K., *The Saphira Architecture for Autonomous Mobile Robots*, Artificial Intelligence and Mobile Robots: Case Studies of Successful Robot Systems, Chapter 9, pp. 211-242, MIT Press, 1998.

[7] Mataric, M.J., *New directions: Robotics: Coordination and learning in multirobot systems*, IEEE Intelligent Systems, vol. 13, no. 2, pp. 6–8, 1998.

[8] Oller, A., del Acebo, E., de la Rosa, J.Ll., *Architecture for Co-operative Dynamical Physical Systems*, 9th European Workshop on Multi-Agent Systems, 1999.

[9] Parker, L.E., *ALLIANCE: An Architecture for Fault Tolerant Multi-Robot Cooperation*, IEEE Transactions on Robotics and Automation,1998, 14(2): pp. 220-240.

[10] Picault, S., Drogoul, A., *The MICRobES Project, an experimental approach towards Open Collective Robotics*, Proceedings of the 5th Internacional Symposium on Distributed Autonomous Robotic Systems (DARS'2000).

[11] Purvis, M., Cranefield, S., Ward, R., *Distributed Software Systems: From objects to agents*, Published in the Proceedings of SE:EP'98, Dunedin, New Zealand.

[12] The Foundation for Intelligent Physical Agents `http://www.fipa.org`

[13] Wooldridge, M., *An Introduction to Multiagent Systems*, John Wiley & Sons Ltd., 2002.