Agent-based Software Integration for a Mobile Manipulator[*]

Patricio Nebot
Intelligence Robotics Lab
Jaume-I University
Castellón, Spain
pnebot@icc.uji.es

Gael Saintluc
ISEN Toulon
Toulon, France
gael_saintluc@yahoo.fr

Benjamin Berton
Ecole Centrale de Nantes
Nantes, France
bberton@ec-nantes.fr

Enric Cervera
Intelligence Robotics Lab
Jaume-I University
Castellón, Spain
ecervera@icc.uji.es

***Abstract** – Mobile manipulation involves mobility and manipulation. For the design of a suitable framework for mobile manipulation it is essential to consider the characteristics of the two systems that constitute the mobile manipulator, the mobile base and the manipulator. The framework should maintain the inherent characteristics for each of the two sub-systems, while at the same time provide a basis for cooperation. In this article, we present a framework suitable for the operation of a mobile manipulator. Such framework is an extension of the Acromovi agent-based architecture. In this framework, the characteristics of each sub-system are preserved, whereas the cooperation is provided by the multiagent system. Also, the framework lets the access to the accessories mounted on each sub-system, thus it is possible to access to the stereo of the arm or to the sonars of the base, making possible the coordination of all the elements to perform a certain task.*

Keywords: Mobile manipulator, integration, agent-based, coordination.

1 Introduction

In the last years, there is an emergent interest in combining the two most important fields of robotics, the control of static manipulators and the navigation of mobile robots. By means of the mobility of the platform, there is an increment of the workspace of the manipulator. Also, having a manipulator on a mobile robot there is an increment of its operational capability and flexibility.

Such systems allow the most usual missions of robotic systems which require both locomotion and manipulation abilities. So, they seem particularly suited for field and service robotics [2].

Many different control systems for mobile manipulators have been developed in the last years. In [7], a control architecture suitable for mobile manipulation is explained. This architecture combines existing techniques for navigation and mobility with a flexible control system for the manipulator arm.

In [3], the authors implement a behaviour-based controller over a mobile manipulator to make feasible that the robot open a door. The key issue for realizing such behaviour is cooperation between robots function sub-systems, such as the locomotion control system, the manipulator control system and the sensor systems.

Other approaches to the control of a mobile manipulator take into account neural networks. In [15], a neural network–based methodology is developed for the motion control of mobile manipulators subject to kinematic constraints.

A task-oriented framework for the dynamic coordination of mobile manipulator systems has been implemented in [10]. The dynamic coordination strategy developed is based on two models concerned with the effector dynamic behavior, and the robot self-posture description and control. The effector dynamic behavior model is obtained by a projection of the robot dynamics into the space associated with the effector task, while the posture behavior is characterized by the complement of this projection.

Finally, other possible approach is to use multiagent system as in [8]. This work presents a multiagent system approach to a service mobile manipulator robot that interacts with human during an object delivery and hand-over task in two dimensions. The agents of the system are controlled using fuzzy control. The functions of the fuzzy controllers are tuned by using genetic algorithms.

In concerning to vision, in [1] a new methodology to achieve vision-based control of a mobile manipulator is presented. This methodology makes possible that the mobile manipulator can reach to a required three-dimensional target position and orientation. The objective is to apply the nonholonomic degrees of freedom represented by two independently driven wheels together with the holonomic dexterity of the on-board arm in order to bring about a desired positioning objective for the arm.

Other work that presents a visual control is [9]. In this project, a high-precision visual control method for mobile manipulators is presented. That method makes efficient use of all the system's degrees of freedom, which reduces the required number of actuators for the manipulator.

[*] **0-7803-8566-7/04/$20.00 © 2004 IEEE.**

Finally, the arm must be able to perform manipulation tasks. In [5], a framework for visually guided manipulation tasks is presented. The complexity of such task determines the precision and freedoms controlled which also affects the robustness and flexibility of the system. The proposed approach results in a system which can locate and grasp a certain object.

The presented paper describes an extension added to the *Acromovi* architecture to manage an arm over a mobile robot. First, the mobile manipulator is described. Then, an overview of the *Acromovi* architecture and the extension are showed. Finally, some tools and applications implemented with such architecture are explained. At last, the future lines are described.

2 Our mobile manipulator

The mobile manipulator used in this is formed by the combination of a Mitsubishi PA10 arm and an ActivMedia Powerbot mobile robot, as in Figure 1.

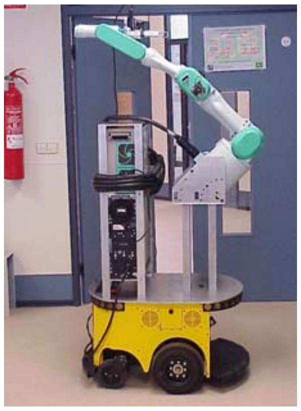


Figure 1. The mobile manipulator: PA10 over Powerbot

The arm has 7 degrees of freedom, and is equipped with a two-finger gripper, a force sensor, some infrared sensors for collision detection and a stereo pair mounted at the end-effector, which consists of two digital IEEE1394 color cameras.

The mobile robot presents at front IR sensors for stairs and holes detection, and around it there are bumpers and sonars. Also, it has an onboard computer and a wireless card, which lets it to communicate with other robots or Internet.

Finally, as can be seen in the photo, the mobile manipulator carries at backwards the controller of the arm and batteries rack to provide the necessary power to the arm.

3 *Acromovi* architecture overview

Acromovi architecture is a distributed architecture for programming and controlling a team of multiple heterogeneous mobile robots that are capable to cooperate between them and with people to achieve tasks of service in daily environments [11].

The mobile robot has already a programming architecture, with native (C/C++) libraries, constituted by two layers. ARIA, the lower layer, and Saphira, the upper layer. But also, the Acromovi architecture is able to subsume any other extra software, like the ACTS (a color tracking library) and the Vislib (a frame grabbing library), as can be seen in Figure 2.

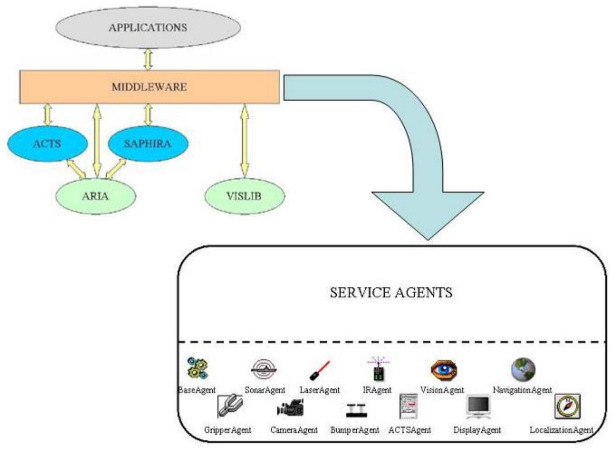


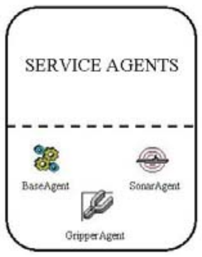
Figure 2. *Acromovi* architecture diagram

Acromovi architecture is a middleware layer between the robot architecture and the applications, which allows sharing the resources of each robot among all the team. This middleware is based on an agent-oriented approach.

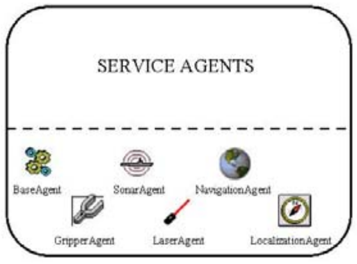
This middleware level has also been divided into two layers. In the lower layer, there are a set of components that access to the physical parts of the robots, as the sonar, the base or the gripper; and other special components that offer services to the upper layer, as the vision, the navigation or the localization. The upper layer is still in development, and comprises a great variety of embedded and supervising agents.

It is important to note that the components that form the lower level of the middleware have also been implemented as agents, due to factors of efficiency, implementation facility and integration with the whole structure.

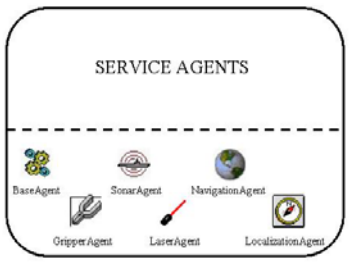
Because of the heterogeneous character of the robots, there are different middleware layers, depending on the configuration of each robot. These middleware layers are generated in execution time according to the elements that each robot has active at the moment of its execution. Therefore, each middleware layer will have active those agents that permit the access to the own components of the robot. Some examples of this can be seen in Figure 3.



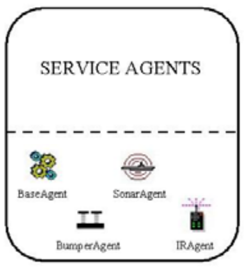
(a) Robot without external systems



(b) Robot with laser system



(a) Robot with camera system



(b) Powerbot

Figure 3. Differences between middleware layers

Over the middleware layer is the applications layer. These applications, to access to the elements of the robots, must communicate with the agents of the middleware, which then access to the bottom layer that controls the robot.

4 *Acromovi* extension

The purpose of this work is to add to the architecture previously described the capability to manage new components, an arm with all its accessories, and make feasible the cooperation between the arm and the mobile base.

As in the case of the Vislib or ACTS, it is necessary to subsume new native libraries in the lower level of the global architecture. These native libraries are the libraries that permit the access to the arm and its accessories. This can be seen in the following figure.

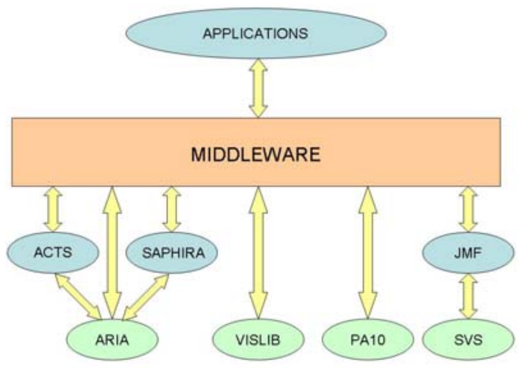


Figure 4. Extended *Acromovi* architecture

The PA10 library is a library to develop application programs for the operation control of the robot, but moreover, is an interface library to ease the operation of all the functions for the motion control of the robot. It is written in C language.

To manage the stereo, we dispose the C++ library – Small Vision System [6]. The SVS consists of a set of library functions for performing stereo correlation. Programs may call these functions to compute stereo results on any images that are available. It is written in C++.

These two libraries need to transfer their functions and information that returns to Java. For this reason, it is used the Java Native Interface (JNI) [14]. In the case of the PA10 library, there is a wrapper that transforms the functions of this library to Java functions. For the vision system, there is a wrapper for the Small Vision System, and it is able to deliver stereo 320x240 color images at 20 frames per second.

The stereo provides information that can be used to control the arm with a Java program. The acquisition of the images need the libraries provided with the cameras. Then these images are used to create the video stream using the Java Media Framework (JMF) [13].

One major advantage of using JMF is that is able to seamlessly transmit video over the network, using the RTP protocol [4]. With JPEG compression (setting the quality factor to 0.5), the system is able to deliver 20 fps of stereo images through a wireless network (802.11b) to the university network, without noticeable loss of quality and response.

Image processing is done in 100% pure Java, keeping pace with the incoming frame rate. Several processing techniques are available:

- Color blob segmentation: up to 32 channels, with multiple blobs on each channel, are available. The centroid, size and bounding box of each blob is computed.

- Edge and corner detection, based on the SUSAN principle [16].
- Line detection, based on the Hough transform [12].
- Motion detection, thorough a simple luminance change detector.
- Regions of interest are used thoroughly, to avoid processing of the complete image.

Other techniques currently being implemented, or planned for the near future, are stereo disparity computation, face recognition, people tracking, and multisensor fusion techniques.

To control these new libraries, it is necessary to include new agents/components to the middleware layer, in more detail, in the lower level. These new components are in charge to serve the petitions of the other agents of the system to the element that they manage.

Thus, the first agent developed is the agent that moves the arm. This agent makes possible the communication between the upper agents and the native library of the arm to move it.

In order to integrate the images of the stereo par with the agent architecture, a new agent has been implemented, the SVSAgent. This agent serves the images that are acquired by the stereo to the system, and offers all the facilities that present the SVS library.

Finally, the JMF agent is capable to create a video stream through the images that serves the SVSAgent. This video frame can be transmitted over the network without an elevate cost due to the jpeg compression. Also, this agent is able to execute some functions of image processing.

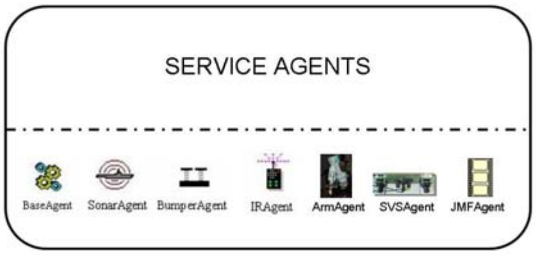


Figure 5. Mobile manipulator middleware

5 Applications

In the following subsections, two applications implemented over the extension of the architecture are described. The first, the arm controller, lets us to move the arm in the two spaces of coordinates, joint and Cartesian. The second, the visual servoing, permits in one hand to move the arm to situate it in a certain distance and position of an object, and in the other hand permits to detect colored objects to control the orientation of the arm.

5.1 Arm controller

This application controls the arm through the network by means of agents. The Control agent is in charge of instantiating the user interface, which is composed of two tabs setting Cartesian in tip or base coordinate and joint motions. This user interface is completely implemented with Java Swing components.

In Figure 4, it is represented the interaction between the different parts that make possible the communication between a user and the arm.

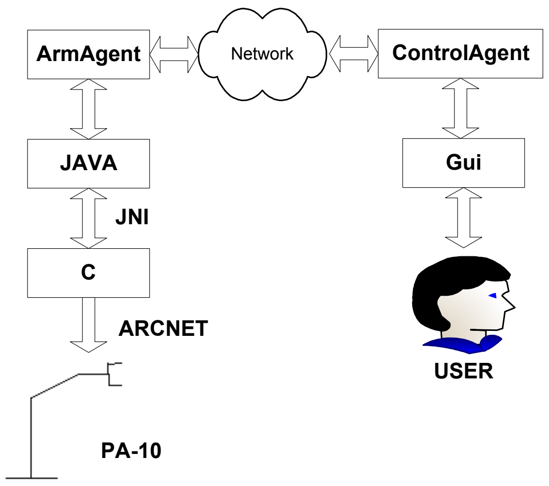
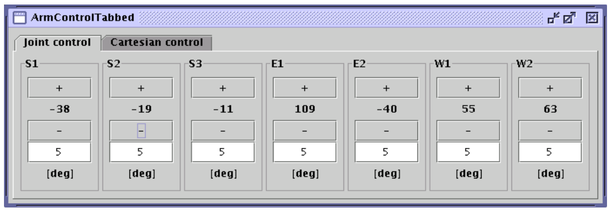


Figure 6. Agents' communication

When the user interacts with the interface, it generates events that are captured by the ControlAgent. This agent, depending on the events, sends different orders through the network to the ArmAgent. Then, the ArmAgent invokes the requested Java function implemented from the native C function provided by the PA10 library via the JNI technology. This function is sent to the arm via ARCnet (*Attached Resource Computer network), which is* one of the oldest, simplest, and least expensive types of local-area network.

The PA10 library communicates with the embedded computer that manages the arm. Communication between the embedded computer and PA10 controller enables either to launch an arm motion or to acquire useful arm information that following the inverse way is showed in the user interface.

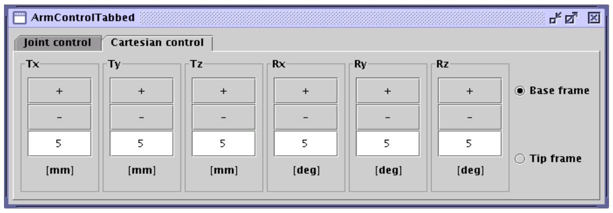
In the following figures, can be seen the two tabs that manages the application. In Figure 5, the joint control tab is showed. By means of this tab, the user can move the arm indicating the relative movement of each joint.



S1	S2	S3	E1	E2	W1	W2
+	+	+	+	+	+	+
-38	-19	-11	109	-40	55	63
-	-	-	-	-	-	-
5	5	5	5	5	5	5
[deg]	[deg]	[deg]	[deg]	[deg]	[deg]	[deg]

Figure 7. Joint control tab of the PA10 arm

In Figure 6, the other tab of the application is illustrated. Using this tab, a user is capable to move the arm in Cartesian coordinates. There are two possible origins of coordinates to execute the arm movements, one is from the base, and the other is from the tip. Also it is possible to specify two different movements of the arm, translational movements or rotational movements.



Tx	Ty	Tz	Rx	Ry	Rz
+	+	+	+	+	+
-	-	-	-	-	-
5	5	5	5	5	5
[mm]	[mm]	[mm]	[deg]	[deg]	[deg]

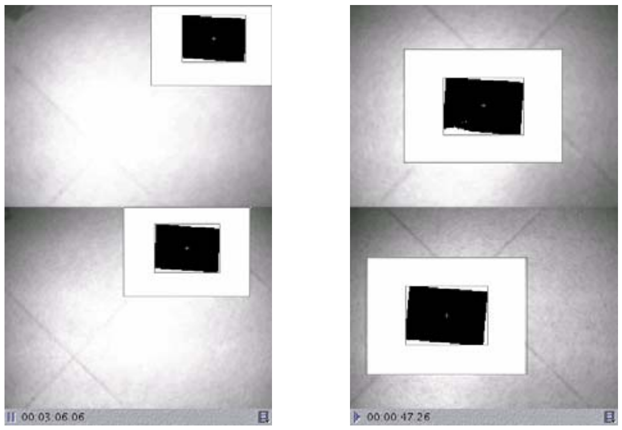
[✓ Base frame]
Tip frame

Figure 8. Cartesian control tab of the PA10 arm

5.2 Visual servoing

In this application, the purpose is move the arm to a certain position respectively an object. This position is that which the object is centered in the image and the arm is a certain distance above the object. To monitoring the process, the two images (left and right) taken from the stereo are displayed in the window.

In the next figure, it is showed the initial and final positions of the process. It is possible to appreciate as in the final position, the object is centered in the image and closer than in the initial position.



(a) Initial position (b) Final position

Figure 9. Visual servoing processing

The program looks for colored objects in a clear background. Then it computes the coordinates of their centers of gravity. The distance between the object and the camera is computed with the disparity between the two images.

With these three pieces of information (coordinates and distance of the object) the arm is controlled to situate it to the needed position. The application communicates with the ArmAgent to move the arm with the three translational movements that this agent offers in the Cartesian space.

In order to control the 7 d.o.f. of the arm, many color objects can be detected. These objects bring the information necessary to control the orientation of the arm that can be modified with the ArmAgent by means of the rotational movement in the Cartesian space.

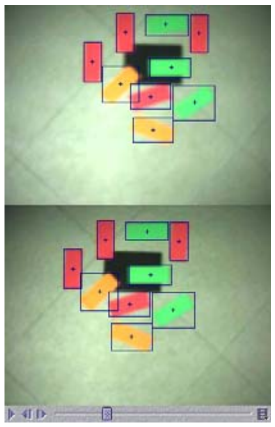


Figure 10. Objects color tracking

6 Future extensions and applications

A first extension for the system is to give it the capability to manage the force sensor of the PA10 arm. This sensor permits to know how much force is doing the arm in a certain moment. This force is produced by the strength that is making the gripper when is pushing any object.

Other extension concerning to the physical elements that dispose the robot is to create the correspondent agent to manage the infrared sensors that wrap the arm in order to avoid collisions.

More complicated extensions could be creating agents to calculate the path that must be followed by the arm to catch an object localized by the stereo.

The most important application for our mobile manipulator is the librarian robot. Due to the arm installed on the mobile robot, this is capable of manipulate books. This robot will be in charge in a library to look for and to extract of the book shelves the requested books, and it will be able to carry them to the users.

Other possible applications can be a robot that can paint a wall or a robot that can navigate by its environment and picks up all the cans that it detect.

References

[1] A. Cardenas, B. Goodwine, S. Skaar and M. Seelinger, “Vision-Based Control of a Mobile Base and On-Board Arm”, *International Journal of Robotics Research*, Vol. 22, No. 7, pp. 677-698, Sep. 2003.

[2] B. Bayle, J.Y. Fourquet and M. Renaud, “Manipulability of Wheeled Mobile Manipulators: Application to Motion Generation”, *International Journal of Robotics Research*, Vol. 22, No. 7, pp. 565-581, Jul. 2003.

[3] B.J.W. Waarsing, M. Nuttin and H. Van Brussel, “Behaviour-based mobile manipulation: the opening of a door”, Proc. ASER’03, Bardolino (Italy), pp. 168-175, Mar. 2003.

[4] C. Perkins, *RTP: Audio and Video for the Internet*, Addison-Wesley, 2003.

[5] D. Kragic, L. Petersson and H. I. Christensen, “Visually Guided Manipulation Tasks”, *Robotics and Autonomous Systems*, Vol. 40, No. 2-3, pp. 193-203, Aug. 2002.

[6] K. Konolige, The SRI Small Vision System, http://www.ai.sri.com/˜konolige/svs/

[7] L. Petersson, M. Egerstedt and H.I. Christensen, “A hybrid control architecture for mobile manipulation”, Proc. IROS’99, Kyongju (Korea), pp. 1285-1291, Oct. 1999.

[8] M.S. Erden, K. Leblebicioglu and U. Halici, “Multi-Agent System-Based Fuzzy Controller Design with Genetic Tuning for a Mobile Manipulator Robot in the Hand Over Task”, *Journal of Intelligent and Robotic Systems*, Vol. 39, No. 3, pp. 287-306, Mar. 2004.

[9] M. Seelinger, J.D. Yoder, E.T. Baumgartner and S.B. Skaar, “High-Precision Visual Control of Mobile Manipulators”, *IEEE Transactions on Robotics and Automation*, Vol. 18, No. 6, pp. 957-965, Dec. 2002.

[10] O. Khatib, “Mobile Manipulation: The Robotic Assistant”, *Robotics and Autonomous Systems*, Vol. 26, No. 2-3, pp. 175-183, Feb. 1999.

[11] P. Nebot, D. Gomez and E. Cervera, “Agents for Cooperative Heterogeneous Mobile Robotics: a Case Study”, Proc. SMC’2003, Washington, D.C. (USA), Vol. 1, pp. 557-562, Oct. 2003.

[12] P.V.C. Hough "Methods and means for recognizing complex patterns," U.S. Patent 3,069,654, Dec 1962.

[13] R. Gordon, S. Talley. *Essential JMF: Java Media Framework,* Prentice Hall, 1998.

[14] S. Liang, *Java(TM) Native Interface: Programmer's Guide and Specification*, Addison-Wesley, 1999.

[15] S. Lin, A.A. Goldenberg, “Neural-Network Control of Mobile Manipulators”, *IEEE Transactions on Neural Networks*, Vol. 12, No. 5, pp. 1121-1133, Sep. 2001.

[16] S.M. Smith and J.M. Brady. “SUSAN - a new approach to low level image processing”, *International Journal of Computer Vision*, Vol. 23, No. 1, pp. 45-78, May. 1997.