Transactions on Information and Communications Technologies vol 1, © 1993 WIT Press, www.witpress.com, ISSN 1743-3517

A mobile robot architecture dedicated to asynchronous events management

N. Pons, S. Delaplace & J. Rabit
Laboratoire de Robotique de Paris, Université Pierre et Marie Curie, Tour 66, 2ème étage, 4 Place Jussieu, 75252 Paris Cedex 05, B164, France

ABSTRACT

Within the L.R.P. mobile robot project ROMEO (French acronym for an experimental autonomous robot) figure 1, we have developed a dedicated software architecture. This architecture is specifically designed to support manoeuvering requirements such as planning control and contacts management with the environment. After a presentation of the navigation problematic in a constrained environment for a non-holonomic robot, we describe our approach and the soft-and-hardware levels consequences deriving from it. Fundamentally we deal with the elaboration of a very flexible and open architecture, which combines functional and behavioral aspects to cope with asynchronous events and to give to the robot reflexivity.

CONTEXT DESCRIPTION

Our aim is to enable ROMEO to achieve complex missions in a structured and partially known environment such as a factory-like environment. In a multi-robot factory, global missions could consist in manipulators supplying, environment surveying,... .

Those missions are divided in navigation tasks which are of two sorts: long distance navigation (displacements in an unconstrained environment) [1], and short distance navigation or manoeuvering tasks (displacements in a constrained environment). At least, each manoeuvering task required for the mission is an action like parking, passing through a door or turning in a corridor and has to be planned and executed in a special way.

A high level system, situated in a ground based station devoted to the different robots managing, defines the mobile robot global mission

according to the robotic cell requests. Using a geometrical and functional world modelling, a global planner creates the mission optimal plan constituted of a set of navigation tasks. It builds a graph where the nodes are representing functional areas and the arcs are denoting the action type. A navigation task is thus expressed as two areas (start and end zones) and an action specification.

Generally speaking, the robot must communicate with the central system from dedicated places. This is often the case in a noisy environment like in a factory plant or a shop floor with machine tools. As a result, the robot has to be entirely autonomous during the execution phase, in other words it has to be doted of a sufficient decisional capacity to automatically generate a local plan when it's dealing with manoeuvers and to manage the asynchronous events occurrences.

Such a description situates the context we are dealing with in our study.

THE GENERAL SOFTWARE ARCHITECTURE

A lot of studies have been dedicated to software architecture design. Different approaches are possible. Some authors are dealing with solutions where the control architecture is based on several control system layers. Each layer achieving a behavior [3],[4],[5]. All of this could be considered as a reactive approach compared to the centralized one where a main controller manages functional modules as planning, perception, execution, control...[6], [7].

Up to now, one could notice a new kind of approach which tends to establish a footbridge between this two directions [8], [9].

Our viewpoint is also to try to combine those two approaches in order to allow, on one hand, a high level strategic planning permitting to perform complex and optimized missions and on the other hand, an implementation of low level reactivity necessary in a non predictable environment.

The global architecture of our robot is represented on figure 1. This architecture relying on four levels contains the different steps from the task level programming of the robot to the execution of the orders permitting the mission realization.

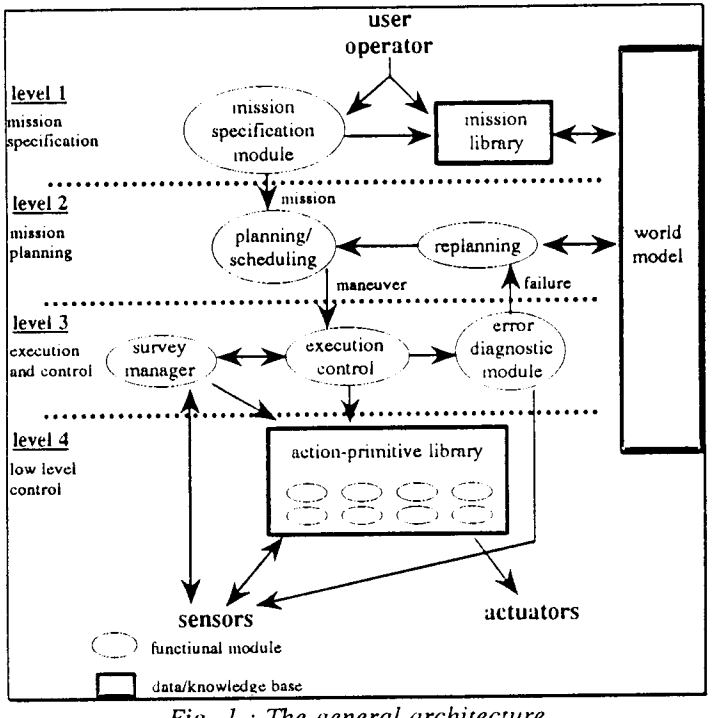

Fig. 1 : The general architecture

At the two highest levels a strategic outline planner [2] elaborates a path based on the optimization of execution bounded criteria. It manages the scheduling of the different tasks, which allows the robot to accomplish its mission.

The two last levels are dedicated to the sensors and actions management. Handling with asynchronous events, it is in charge of dealing with the manoeuvers tasks realization.

These levels are located on board while the latter are supported by a ground based station (SUN workstation). The robot is radio-linked to the off-board system. In this paper we will focus on the lowest levels.

A MANOEUVERS ORIENTED APPROACH

The main criteria taken into account in our approach are related to the robot safety and reflexivity and to a great independence from the high level system. According to these criteria, the adopted strategy consists in simplifying and formalizing objects, such as trajectories or actions, in the aim to ease their manipulation in a dynamic way with respect to the tasks under real time execution. Receiving a manoeuver order, the on-board system use pre-defined structures that it has to instantiate to generate

trajectories, local plans, reflexes, reactions,...To create these objects, the system uses bases of rules which are adapted to the common situations avoiding fastidious analysis handled by the expert. Finally, to obtain reflexive behaviors, an autonomous robot must be able to handle with asynchronous events like unexpected obstacles or contact occurrences.

So, to increase reflexivity and independence, we have used an object representation. Each manoeuver is considered as a set of objects called "action-object" which can be manipulated and instantiated in the appropriate form. Formalizing the manoeuvering tasks in such a way presents several advantages like :

- an easy programming at the highest level and a great independence from the strategic out line planner (data flow between the global planner, situated in a ground based station, and the robot is reduced to a manoeuver order and some parameters, which transmission is done through a serial link). After reception, the robot generates automatically all the requirements needed for planning and surveying the execution. It uses bases of rules instantiated on line according to some environmental criteria.
- introduction of reflexive modes. By itself the reflex is an "action-objects" and it is manipulated in the same way as the previous ones.
- possibility of reorganizing dynamically the actions scheduling. It allows to manage asynchronous events and moreover to consider some degraded modes (lack of some sensors,...).

The trajectory the robot has to follow is not a priori given, but, constituted of a succession of straight lines generated during the creation of the local plan. These characteristic straight lines are associated to each maneuvering tasks. Straight lines are either issued from mathematical models (analytical lines) or from a task like "ultrasonic wall following" (contextual lines).

To compensate sensor measurement errors or overtracking during trajectory following, we take into account the contacts, which can occur between the robot and the environment. As soon as a contact is detected and identified the robot react in a reflexive way. First, it leaves the contact (we call here reflex the action to leave the contact) and, then, elaborates a reaction in order to find an issue for the task in execution. For that, we have provided the robot with a tactile bumper permitting to detect and locate a contact.

THE EXECUTION CONTROL STRUCTURE

We will now focus on the mid level of our architecture : the execution control (fig.2). This level is dedicated to the actions and sensors management.

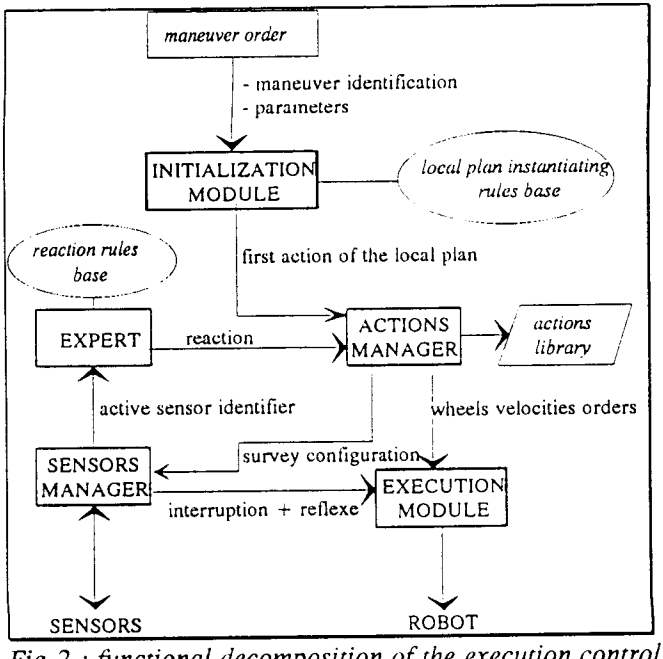

Fig.2 : functional decomposition of the execution control

Using "action-primitives" provided by the lowest level the on-board system has, as a first task, to create the local plan of the manoeuver. This plan is elaborated by the *Initialization Module(I.M.)* which uses an associated base of rules : the local plan instantiating rules base.

After reception of the plan, the *Actions Manager module(A.M.)* computes the wheels velocities orders values owing to low level control laws ensuring the required motions execution. These values are then transmitted to the *Execution Module(E.M.)*.

Low level control must ensure the correct execution of a desired behavior of the robot. This is a prerequisite to obtain good performances system especially from high level considerations. Moreover, a redundancy in the low level controls will increase the robustness of the system and allow to manage some degraded modes.

According to that, we have installed an "action-primitive" library which contains all the actions that the robot may need to execute its mission, and the different ways to do it.

An action-primitive is the specification of an action like turn, move for-or-backward or follow a straight line and the way the control is done. An action can be executed in an open loop or in a closed loop way. In the last case, we have chosen to use state feedback or fuzzy control. To increase the performances of the robot control, it is possible to add as much action-primitives as we need.

Beside this library, a list of criteria orients the system to the best action-primitive for the task in progress. These criteria are issued from a comparative study between different control laws (under sensors actions), applied to characteristic configurations occurring during a maneuver execution. Moreover, this choice is made according to the sensors availability to combine material constraints with performance factors.

These modules are sufficient to allow the robot to perform a maneuver if no asynchronous events occur. Because of the above considerations, it is inherent to this level to manage the asynchronous events. Only external events, interactions between the robot and its environment, are for the moment considered in our application. These asynchronous events as unexpected obstacles detection or contact occurrences, are known but their occurrences are undefined in time.

The asynchronous events management is taken in charge by two modules, the *Sensors Manager(S.M.)* and the *Reaction Expert(R.E.)*.

As soon as an event has been detected, the sensor manager sends on one hand, an action interruption to the Execution Module and, according to the type of event, a reflex. On the other hand, this module provides the event identification to the *Reaction Expert*. After the reaction elaboration, the Expert module sends it to the Action Manager.

When the robot has detected an obstacle , the robot uses an obstacle avoidance primitive until it reaches the desired trajectory again. But, when it has collided with an obstacle, the Reaction Expert determines which reaction the robot has to execute to accomplish the desired task.

The modules of the proposed architecture are manipulating "action-object". An action-object is a class of action whose fields are :

- the action specification
- the controlled point
- the End Condition (E.C)
- the survey configuration
- the next action-object address.

The number of fields is not limited and can be increased for a greater precision. The last field (next action address) permits either to link action-objects together as for the local plan or, for the asynchronous events management, to orient the execution to a reaction and then to return to the interrupted plan.

The local plan of a maneuver contains four phases:

- the approach phase
- the active phase
- the final phase
- the End Message return

The first three phases are maneuvering phases and consist of action-object. The last phase provides to the global planner a message containing the status of the robot, that is to say if the maneuver has been accomplished or not.

This structure contains three reactivity levels :

- the reflex
- the reaction
- the partial re-planning

The reflex loop is the lowest level loop corresponding to a safety action independent from the action in progress. The sensors data are processed by the Sensors Manager which directly generates robot execution orders. The response of the robot is here immediate.

The reaction loop includes the Expert which determines an action according to the context of the action.

Finally, in the cases where the expert does not find solutions to allow the robot to accomplish the manoeuver, a re-planning request is sent by the Error Handler to the global planner.

EXPERIMENTAL RESULTS

To illustrate the control system operations, we have chosen an easy mission including "the through a door" configuration and to turn in a corridor as manoeuver examples. Before describing the different steps of this mission, we describe the current implementation status on our robot ROMEO.

ROMEO has two independent rear driving wheels and two free wheels in front. Currently, its on-board sensors are :

- a set of ultrasonic sensors situated on its periphery[10]
- an odometer
- a CCD camera mounted on a turret with two degrees of freedom
- a tactile bumper permitting to detect and locate contacts.

The modules managing the ultrasonics, the odometer, the tactile bumper and the modules dedicated to the actuators control are implemented on specialized carts.

The global mission consists in our example, to navigate from a given corridor area A1 to another room area A2. In this case, the global plan, constituted by a succession of navigation tasks is transmitted to the robot on the following form :

1. turning in a corridor (from A0 to A1)
2. a corridor wall following using ultrasonic sensors (from A1 to the door 1 detection)
3. throughout the door passing (environmental parameters)
4. a trajectory tracking (up to A2)

Fig.3 : global plan of the mission

The tasks 2 and 4 are long distance navigation tasks and can be directly executed by the robot using the appropriate action-primitive of the library. However, the first and the third one are considered as manœuvers and have to be planned in a special way. The local plan of the manœuvers is created from the parameters provided by the global planner and each action-object field of the plan is instantiated by the Initialization Module.

We have chosen two examples to show the different robot behaviors when the robot collides with the environment. The "through a door" example illustrates the reaction elaboration and execution and the turn in a corridor the reflex behavior of the robot.

The local plan provided by the I.M. and permitting to pass through the door1 is the following one :

1. approach phase
Action : ultrasonic wall following
Controlled point M (middle point of the wheels axis)
End Condition : ultrasonic discontinuity
Survey configuration : active bumper & ultrasonics
Next action address : address of action-object2
2. active phase
Action : straight line L1 forward tracking
Controlled point M (middle point of the wheels axis)
End Condition : odometric condition
Survey configuration : active bumper
Next action address : address of action-object3

3. final phase
Action : straight line L2 forward tracking
Controlled point N (point in front of the robot)
End Condition : odometric condition
Survey configuration : active bumper & ultrasonics
Next action address : address of action-object4
4. end phase
Action : activate the end task & return a message to the global planner

Fig. 4 : "through a door"configuration local plan

The real trajectory executed by the robot is represented on figure 6 and corresponds to the execution of the three last phases of the "through a door" configuration local plan, when no asynchronous events occurs.

An other experimentation demonstrates the expert intervention when a frontal contact occurs during the active phase execution (action_object2) of this samemanoeuver. In this case, a reflex is not necessary and a reaction is directly created by the expert (Fig. 5) and transmitted to the robot via the Action Manager. In the reaction itself is contained the next action address permitting the robot to go on with the manoeuver after the reaction execution.

5. reaction phase
Action : straight line L1 backward tracking
Controlled point M (middle point of the wheels axis)
End Condition : odometric condition
Survey configuration : active bumper
Next action address : address of action-object2

Fig.5 : reaction on line created by the Expert Module

The trajectory the robot is realizing during the execution of the manoeuver is represented on figure 7. As the contact occurs, the robot executes the reaction and then the interrupted active phase and the final phase of the local plan.

In some case it is not necessary to create a reaction and a reflex is sufficient to permit the realization of the action as shown in the corridor example. This time, the robot turns in a corridor using trajectory tracking action-primitives. Figure 8 and 9 are representing the real trajectories executed by the robot, on one hand, when no contact occurs and, on the

other hand, when the robot has collided with an unexpected obstacle. The reflex depending only on the nature of the contact is directly provided by the Survey Module without the need of the expert.

CONCLUSION

In this paper we have presented the approach we have retained to specify our soft-and-hardware architecture. Our goal is achieved, i.e., the robot, as well unlinked to a central system, is able to perform complex and optimized missions in an partially known environment. Missions including short and long navigation tasks have been realized with success. In the near future we will optimize the maneuvers execution and increase the low level robot abilities.

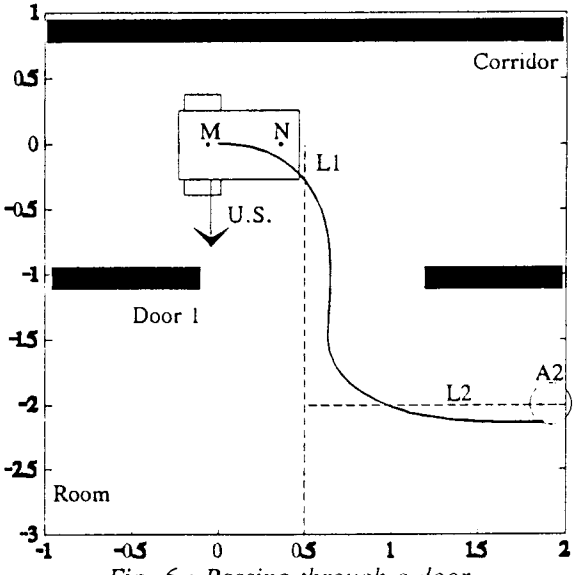

Fig. 6 : Passing through a door

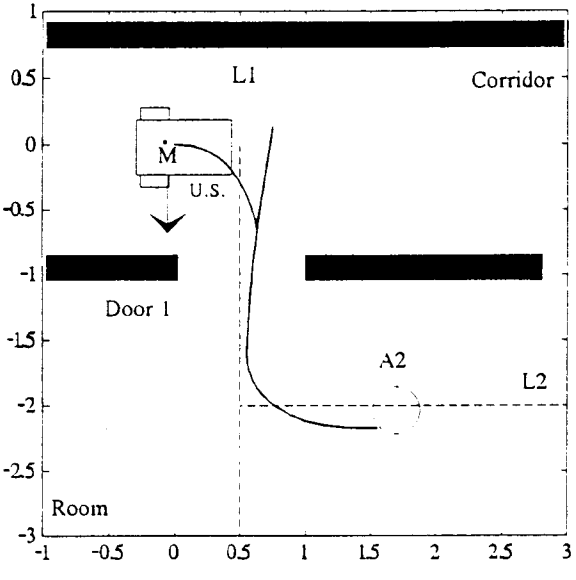

Fig. 7 : Robot reaction

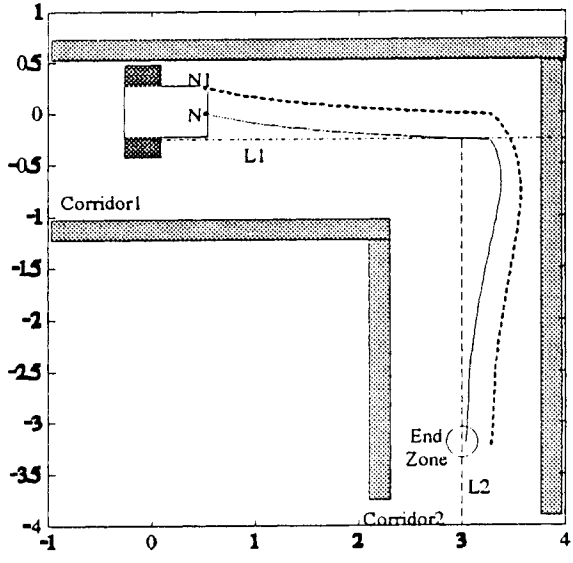

Fig. 8 : turning in a corridor

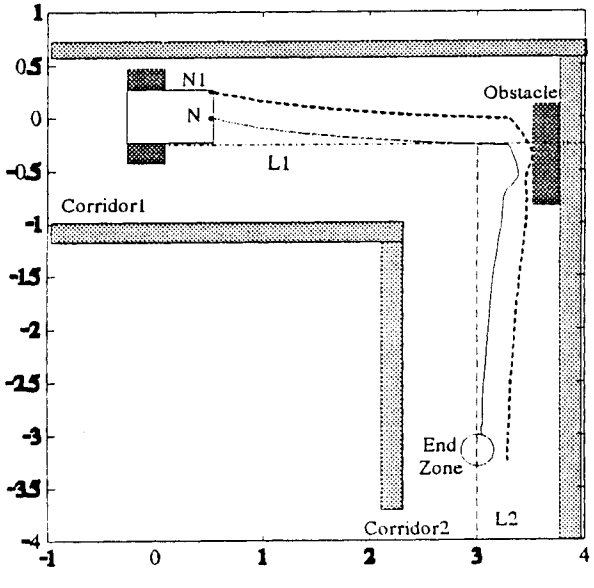

Fig. 9 : Robot reflex

REFERENCES

1. S. Delaplace, P. Blazevic, J.G. Fontaine, N. Pons, J. Rabit. Trajectory tracking for mobile robot.
   EURISCON'91
   ROBOTICS SYSTEMS
   Advanced techniques and applications
   Edited by S. G. Tzafestas
   pp 313-320,1991
2. V. Schaeffer, A. Mauboussin. Task modeling for planning an autonomous mobile robot.
   EURISCON'91
   ROBOTICS SYSTEMS
   Advanced techniques and applications
   Edited by S. G. Tzafestas
   pp 287-295,1991
3. Rodney A.Brooks. A robust layered control system for a mobile robot.
   IEEE transaction on Robotics and Automation,
   RA-2(1):14-23,1986
4. Luis Correia, A.Steiger-Garçao. A reactive architecture and a world model for an autonomous underwater vehicle.
   IEEE/RSJ International workshop on intelligent Robot and Systems IROS'91.
   pp 1472-1476,1991
5. David W.Payton. An architecture for reflexive autonomous vehicle.
   Proceedings of the 1986 IEEE International Conference on Robotics and Automation
   pp 1838-1845,1986
6. Y. Goto, A. Stentz. The CMU System for mobile robot navigation.
   Proceedings of the 1987 IEEE International Conference on Robotics and Automation
   pp 99-105,1987
7. J. Bares, M. Hebert, T.Kanade, E. Krotkov, T. Mitchell, R. Simmons, W. Whittaker. AMBLER, An autonomous Rover for Planetary Exploration.
   Proceedings of the 1989 IEEE International Conference on Robotics and Automation
   pp 18-25,1989
8. Fabrice R. Noreils, Roland Prajoux. From planning to execution monitoring control for indoor mobile.
   Proceedings of the 1991 IEEE International Conference on Robotics and Automation

pp 1510-1517,1991

9. Wonyun choi, J.C. Latombe. A reactive architecture for planning and executing robot motion with incomplete knowledge.
IEEE/RSJ International workshop on intelligent Robot and Systems IROS'91.
pp 24-29,1991

10. P. Blazevic, S. Delaplace, J.G. Fontaine, J.Rabit. Mobile robot using ultrasonic sensors. Study of a degraded mode.
Robotica 1991,vol. 9
pp 365-370,1991