Draft Version
Published in IEEE Trans. on Sys. Man and Cybernetics B, vol.36(5) 2006

# A Control Architecture for Human-Robot Integration: Application to a Robotic Wheelchair*

Galindo C., Gonzalez J., and Fernandez-Madrigal J.A.
System Engineering and Automation Department
University of Málaga
Campus de Teatinos, 29071 Málaga, Spain
Email: {cipriano,jgonzalez,jafma}@ctima.uma.es

## Abstract

Completely autonomous performance of a mobile robot within non-controlled and dynamic environments is not possible yet due to different reasons including environment uncertainty, sensor/software robustness, limited robotic abilities, etc. But in assistant applications in which a human is always present, she/he can make up for the lack of robot autonomy by helping it when needed. In this paper we propose *human-robot integration* as a mechanism to augment/improve the robot autonomy in daily scenarios. Through the human-robot integration concept, we take a further step in the typical human-robot relation, since we consider her/him as a constituent part of the human-robot system which takes full advantage of the sum of their abilities. In order to materialize this human integration into the system, we present a control architecture, called *ACHRIN*, that enables her/him from a high decisional level, i.e. deliberating a plan, to a physical low-level, i.e. opening a door. The presented control architecture has been implemented to test human-robot integration on a real robotic application. In particular, several real experiences have been conducted on a robotic wheelchair aimed to provided mobility to elderly people.

*This work was supported by the Spanish Government under research contract CICYT-DPI05-01391.

## I. INTRODUCTION

Robotic presence in our daily life is still limited. The main reason is that full autonomous performance of mobile robots within real scenarios is not possible yet, mainly due to difficulties in coping with highly dynamic environments, treating with uncertainty, sensor and/or software robustness, limited robotic abilities, etc. However, in certain applications this lack of autonomy may be tolerated since the robot works closely to people (the so-called *assistant robots*). This is the case of one of our mobile robots, called *SENA* [36] (see figure 1), a robotic wheelchair intended to facilitate mobility to impaired and elderly people.

SENA exhibits a high degree of autonomy, but it can not still operate completely by itself within human environments, such as office buildings. Its work (mainly intelligent navigation) may occasionally fail due to the inability of the vehicle to perform some actions such as open a closed door, call a lift or its failure to detect unmodelled or unexpected situations (due to a limited performance of its sensors). These problems could be a serious limitation in some applications, but in our case, a person (the wheelchair user) is closely ”connected” to the vehicle and thus could help the robot. Notice that if the human help was only demanded very occasionally, the user would consider the mobile robot as almost autonomous.

In this paper, we consider that in the particular case of assistant robots, the user can *physically* help the robot by both extending robot abilities (for example to open a locked door) and improving them (the human can perform a robust and reliable navigation by manually guiding the vehicle). Moreover, she/he can participate at several abstraction levels of the system, ranging from a *high level*, i.e. advising a route to arrive a destination, to a *low-level*, reporting information as an extra sensor (i.e. the human can work detecting close obstacles).

The contribution of this paper is the design and implementation of this *human-robot integration* idea into a particular robotic system (the SENA robotic wheelchair), which permits a person to extend/improve the autonomy of the whole system by participating at all levels of the robot operation, from deliberating a plan to executing and controlling it.

In the robotic literature, human participation in robot operation is not a new idea. Some terms have been coined to reflect this, such as *cooperation* [15], [40], *collaboration* [12], [11], or *supervision* [12], [38]. In all these cases the human is an external actor with respect to the robotic system, who can only order robot tasks or supervise its work. In this paper, we take a further step by *integrating* the human into the system, considering her/him as a constituent part of it.

To make possible the human integration claimed in this work, we have identified and addressed the following goals:

- *(a) To manage knowledge about the human physical abilities*. The system must be aware of the physical abilities that the human can carry out in order to decide how and when to use them. For example, in our robotic application, the wheelchair user can manipulate objects (open a door or call a lift) and perform manually navigation through a joystick, while the vehicle can perform navigation via several algorithms. Thus, human integration may provide new abilities, and therefore, the robotic system must be able to consider them when

Fig. 1. The robotic wheelchair SENA. It is based on a commercial electric wheelchair which has been endowed with several sensors (infrared, laser, and a CCD camera). Wheelchair motors as well as sensors are managed by a microcontroller communicated to a laptop computer via USB.

planning/executing a task. In our current scheme we implement a simple selection process to choose abilities from the available human-robot repertory, consisting of selecting human abilities only when the robot is not able to perform the same actions.

- *(b) To consider the human perceptual abilities*. Robot sensors can not capture reliably enough either the highly dynamic nature of real scenarios or some risky situations (for instance, glass doors are not detected by laser sensors), which may cause robot crashes. The integration of a human into the robotic system may extend the robot autonomy by permitting him/her to work as an intelligent sensor reporting or predicting dangerous situations.
- *(c) Detection of execution failures*. The robotic system must detect whether the execution of the current action has failed. The recovery action may include the human: for example a navigational error can be detected, inquiring the human help to drive the vehicle to a safety location. For that purpose, the human could also

improve the robot's capacity of detecting those failures.

- *(d) High-level communication*. Finally, active human presence within the robotic system obviously requires a high-level communication mechanism enabling the user and the robot to interact in a human-like manner, i.e., using natural language, with human concepts which involves symbols such as ”door”, ”room”, ”corridor”, etc.

For addressing these goals, we have designed a specific control architecture, called *ACHRIN* (figure 2) that copes with the above mentioned issues: (a) it enables the task planning process to consider human actions, and permits the user to sequence and execute them at the lower levels of the architecture; (b) it permits the user to verbally report collision alerts at any time; (c) it both checks regularly the performance of the vehicle and enables the user to report failures. Finally goal (d) is approached by the use of a hierarchical and symbolic model of space which serves as a suitable interface with the human cognitive map [9], as well as by commercial speech recognition and voice generation software [41], [42].

The human integration is materialized by designing every component of ACHRIN through a common structure called *CMS* (*common module structure*), which treats humans as software algorithms that provide certain results after being executed. Each CMS groups and manages *skills* (carried out either by the human or the robot) which are aimed to achieve a similar goal. For instance, one of the CMS of ACHRIN is the *Navigational CMS*, that entails different skills to move the vehicle between two locations: a variety of robotic skills, like reactive navigation or tracking a computed or recorded path, and a human skill consisting of manually guiding. These skills are invoked and managed by ACHRIN in the same manner without distinction of the agent that performs the action, human or robot. The human integration we have achieved with ACHRIN enables the user from performing low-level navigation tasks like manoeuvering in complex situations, to performing high-level tasks, like modelling symbolically the workspace.

In the rest of the paper, section II reviews some works related to robotic architectures that support the human presence and participation in the robot operation, while our proposed architecture for human-robot integration is presented in section III. Following sections detail the human integration achieved by ACHRIN at different levels: section IV is devoted to the high-level human-robot integration, section V describes the integration at an intermediate level, while the low-level integration is presented in section VI. Section VII describes different real experiences on our robotic wheelchair aimed to test the suitability of the proposed human-robot integration. Finally, some conclusions are outlined.

## II. RELATED WORKS

Human Robot Interaction (HRI) has been largely treated in the robotics literature from different perspectives. [38] proposes five different human roles in robotic applications (from supervisor to bystander) which cover most of the found approaches. The most common human robot interaction is to consider the human as a robot's supervisor (*supervisory control* [39], [21]). This implies that tasks are performed by the robot under the supervision of a human instead of the human performing direct manual execution.

In the teleoperation area, *collaborative control* [12], [11] can be applied, which is a particular instantiation of supervisory control. Through collaborative control, robots and human dialogue to decide the actions to be carried out by the machine. This relation between human and robot improves the robot operating capacity, but it prevents the human to physically act when the robot is not capable to continue its plan, for example when it must pass through a closed door.

The concept of *cooperation* (see for example [15], [40]) is also well spread within the robotic community. Through human-robot cooperation, humans can perform some coordinate tasks with machines, adopting different roles ranging from coordinator, where the human (also typically an expert) only supervises the robot operation, to a role of a robot partner, where human and robot work independently to achieve a common objective. Nevertheless, these approaches only consider humans as external agents of the robotic system, but not as a constituent part of an unique human+robot system that can take full advantage of the sum of their abilities.

There is a variety of robotic architectures aimed at supporting human-robot relations of the type described before. The general trend, also followed in our work, is to consider a three-layered software architecture with the typical *deliberative, intermediate, and reactive* tiers. But what makes ACHRIN different from the rest is the level at which the human interacts with the robotic system. Most of robotic architectures implements human-robot interaction at the highest level, that is, they rely on the human planning capabilities while low-level operations, i.e. navigation, are carried out by the machine [29], [30], [28], [33]. A few works considers the human participation at the intermediate level [19], [23], or at the reactive layer [1]. But up to our knowledge, no work has approached a general framework to consider (integrate) the human participation into the robotic system at all levels of abstraction. The closest work to ours, presented in [34], [32], also focusses on the human-robot interaction for robotic wheelchairs. They also propose a human interaction at all levels of the architecture, but in a restrictive manner, without considering the wide range of abilities that the human can offer such as perception and manipulation abilities, world modeling, task-planning, etc. One of the novelties of our approach is the integration into the architecture of a cognitive spatial model that seems similar to the one used by humans for space modelling [18], [10], [25]. The use of this common cognitive model enhances both the human-robot interaction and the robot operating capacity. Related works on human participation into the robotic system to create the common cognitive model can be found in [9], [24].

The term ”human integration” has been previously used in [1] in the same terms as in this paper, but only considering human abilities at the lowest level of the architecture. This work, as all approaches classed as *shared control* systems [27], [43], combines at the same time human and robot commands to perform low-level tasks, (i.e. control the velocity of a vehicle), however, it lacks for mechanisms to allow the human to take full control of the robot which become necessary in assistant applications.

Other works implement the so-called *adjustable autonomy* in which machines can dynamically vary their own autonomy, transferring decision making control to other entities (typically human users) in key situations [4], [6], [7], [37]. Since a human can take decisional tasks, ACHRIN can be seemed as an adjustable autonomy system, but in addition, it enables the human to perform physical actions.

Finally, the *Human Centered Robotics* (HCR) concept has emerged to cope with some of the specific requirements of robotic applications within human environments [6], [22], [31]. Among the different questions considered by the HCR paradigm, two important issues are *dependability* and *human-friendly interaction*. Dependability refers to physical safety for both people and the robot, as well as to operating robustness and fault tolerance. Also, HCR considers human-friendly communication, which implies the capability of easily commanding the robot as well as reporting execution information in a proper human way. The work presented in this paper fits into the HCR paradigm since improving mechanisms for human integration into the robotic system (as it is aimed by ACHRIN) also strives for robot dependability and human-friendly interaction.

## III. ACHRIN OVERVIEW

The ArChitecture for Human-Robot Integration (ACHRIN) is based on a hybrid scheme made up of a number of elements, called *modules*[1], which are grouped into three layers (see figure 2):

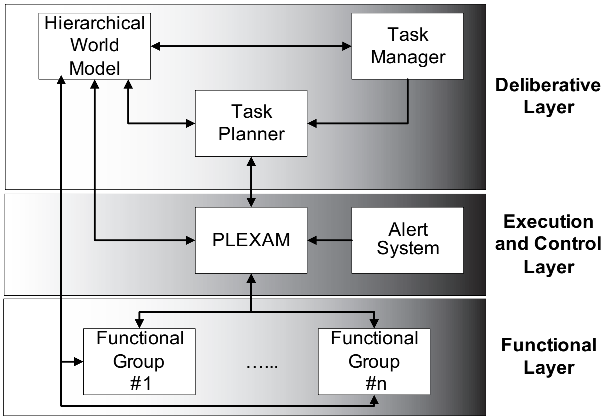


Fig. 2. A general view of ACHRIN. Broadly, it can be considered as a hybrid robotic architecture. However, it does not fit strictly into that typical hierarchical arrangement. For example, the World Model module can be accessed by modules of all the layers.

- *The Deliberative layer* maintains and reasons about a symbolic and hierarchical representation of the environment. Such an internal world model is used to produce plans (sequence of human/robot actions to solve a goal) as well as to facilitate human-robot communication (more in section IV).
- *The Execution and Control layer* sequences and supervises the execution of plans taking into account the information collected from the functional layer and the robot's sensors (refer to section V). According to such information, it may tune the behavior of certain modules, i.e. reducing the vehicle speed when dangerous situations are identified.
- *The Functional layer* comprises a number of groups of skills, called *functional groups*, which physically perform actions, like navigation, manipulation, etc. (see section VI). Each functional group may entail different ways to accomplish a particular type of action[2]. For example, the robot can traverse between two spatial points either by a reactive algorithm, by tracking a computed path, or by the user manual guidance.

---

[1] This architecture can be enhanced through a multi-agent approach. A first steps towards a multi-agent version of ACHRIN is presented in [2].

[2] In our robotic wheelchair application, manipulation actions are carried out exclusively by the human.

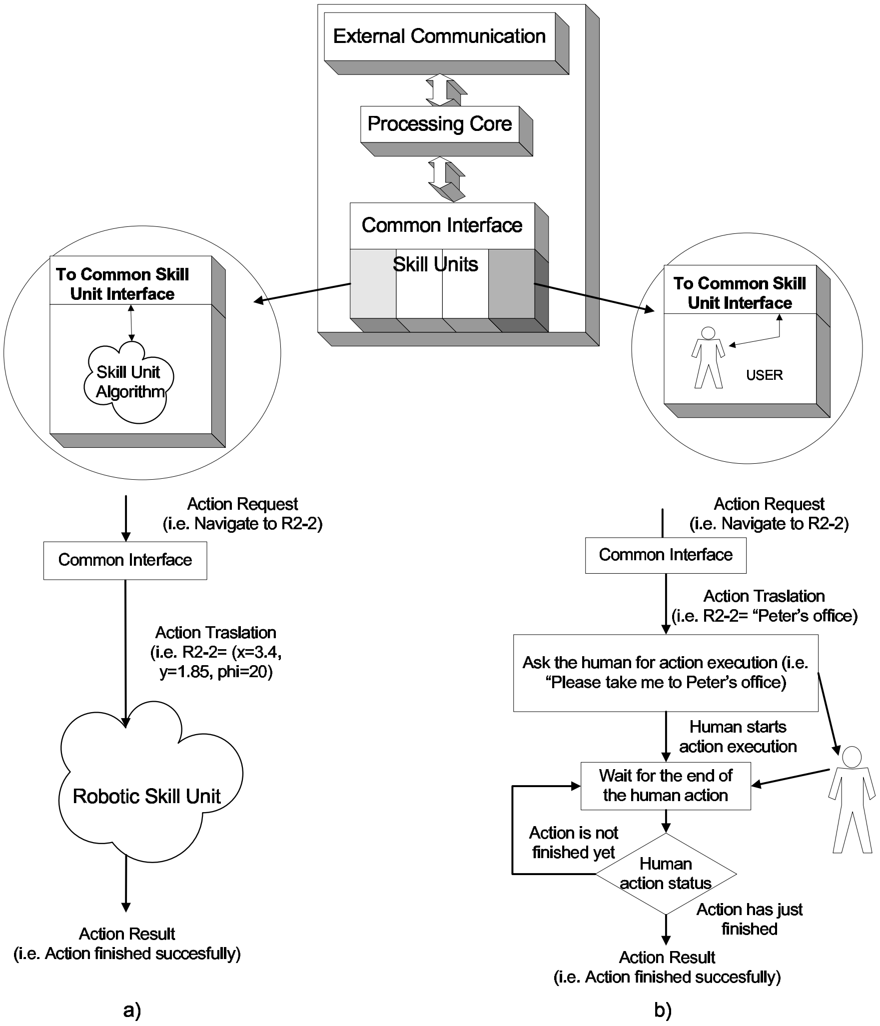


Fig. 3. The common module structure (CMS). (Center top) All architecture components are designed using this structure. The number of skill units contained into each module is variable and depends on the human and robot abilities that the CMS provides. (Bottom a) A representation of a robotic skill unit. (Bottom b) A human skill unit. Notice that the human unit requires the natural description of the destination (”Peter office”), internally represented by the robot as *R2-2*, while the robotic unit only requires its geometrical or topological position.

To support the all-level human integration we claim in this paper, we have used the *common module structure* (CMS) shown in figure 3 for all the modules in the architecture. The CMS integrates human and robot abilities through the so called *skill units*. Each CMS may contain a variety of (human or robotic) skill units that materialize a particular type of ability, like producing a plan, checking for risky situations, moving between two locations, manipulating objects, etc.

In a deeper description, the elements that constitute the CMS are the following:

- *Skill Units*. Skill units execute the action that the module is intended to carry out. Both robotic and human skill units return to the processing core a report, indicating whether they have executed correctly or not.

- *Skill Unit Common Interface*. Although units within the same module of the architecture carry out the same action, they may exhibit differences. For example, the plan to achieve a goal may include a navigation action to a symbolic location, like ”Go to R2-2”. To execute such an action, a robotic unit may need the geometric position of the spatial concept ”R2-2”, let say $(x = 3.4, y = 1.85, \phi = 20)$, but a human skill unit would rather need its linguistic label, i.e. ”Peter’s office”. The Common Interface component of the CMS retrieves from the *World Model module* the information needed in each case[3].
- *Processing Core*. It receives action requests, i.e. ”navigate to a place”, and invokes the corresponding skill unit to execute them. When no extra information is provided, the processing core chooses a robot skill unit to accomplish the requested action following a certain selection policy, i.e. the one with the highest level of past success, that is learning from experience (see a preliminary work in [2]). The processing core is also in charge of receiving and communicating to the rest of ACHRIN’s modules the results of the skill unit execution.
- *External Communications*. This element encapsulates two mechanisms to communicate different modules of the architecture: *client/server requests* and *events*. The client/server mechanism is a one-to-one communication mechanism that allows modules to request/provide action execution, information, etc., while events are a one-to-many communication mechanism, that is, a signal which is ideally, simultaneously communicated to all modules. In our implementation, events are used to broadcast system alerts, like collision risks or battery failures, to every module of ACHRIN.

Next sections detail the architecture tiers and their modules.

## IV. CONCEPTUAL HUMAN-ROBOT INTEGRATION

The deliberative layer of ACHRIN is in charge of maintaining and manipulating a symbolic model of the environment. Such a model is needed for deliberative processes (planning), but it can also serve for improving human-robot communication. In this work we rely on a hierarchical model of the space which has demonstrated its suitability for the ease of communicating with humans [9].

The modules entailed in this layer are: the *Hierarchical World Model* module, that holds the symbolic model, the *Task Manager*, that manages goals requests to be planned and executed by the robotic system, and the *Task Planner*, that generates plans to attain the requested goals.

### *A. The Hierarchical World Model Module*

Apart from the use of suitable voice interfaces [41], [42], to approach a high level communication (goal (d) in the introduction), it is needed to endow the robot with a world representation compatible with the human internal representation of space. For that, we have extended the well-known approach of the ”topological map” [35] with a hierarchical structure.

---

[3] This is the reason of the pervasive interconnection of almost all modules of the architecture to the World Model module. The Alert module is the unique exception since its work is purely subsymbolic.

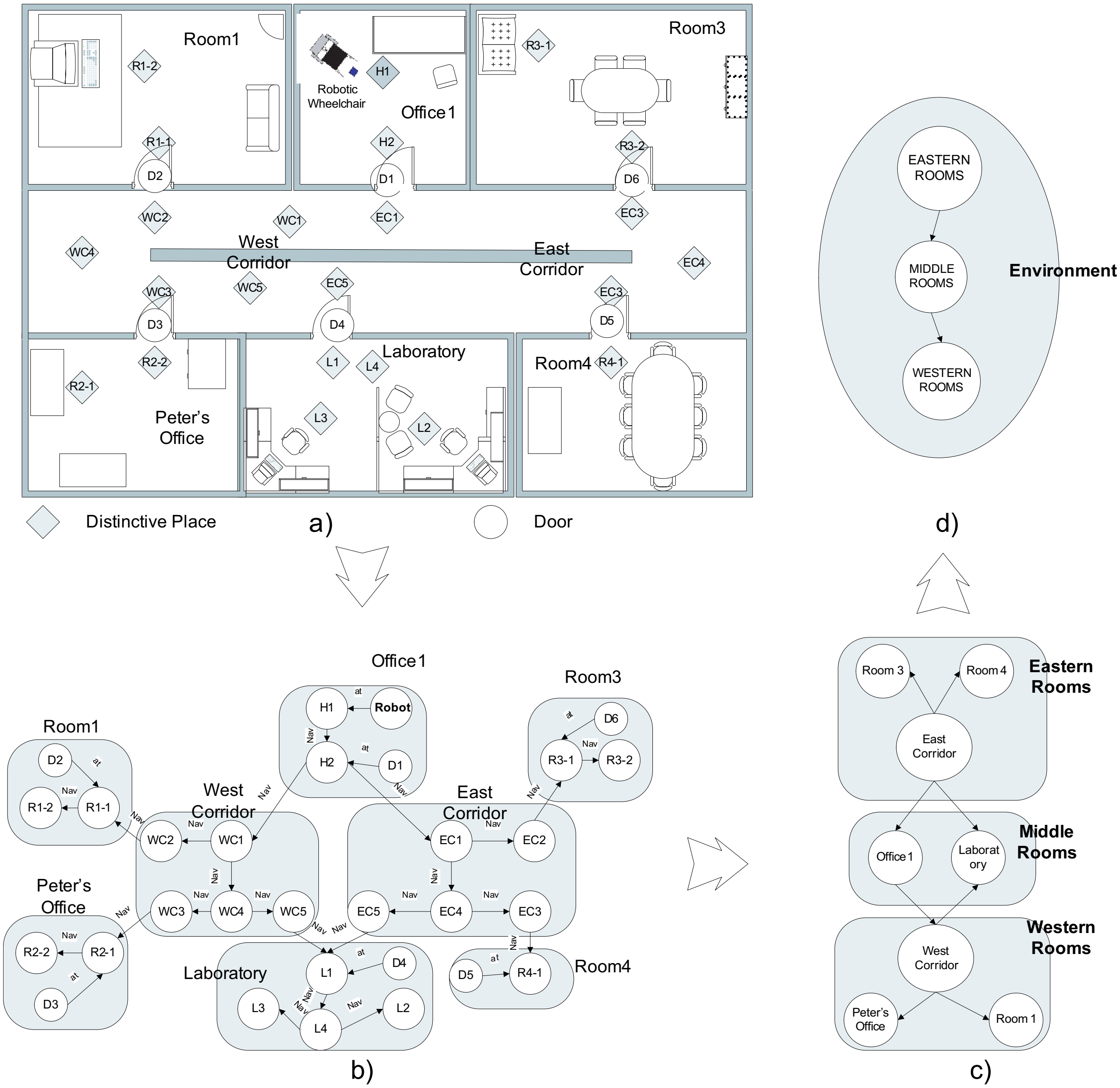


Fig. 4. An example of a hierarchy of abstraction. (a) A schematic plan of a real environment. Distinctive places for robot navigation are marked with small rhombuses. (b) Ground level: topological map of distinctive places. (c-d) Upper levels of the hierarchy. Gray-shaded regions contain the set of nodes which are abstracted to a common spatial concept. Linguistic labels can be attached to spatial concepts to improve human-robot communication

It is stated in literature that humans widely use a mechanism called *abstraction* to store and manage efficiently the huge amount of information that emerges from their environment [16], [17], [25], [26]. This serves to reduce the amount of information considered for coping with a complex and high-detailed world: world elements are grouped into more general (abstract) entities, which can be abstracted again. Thus, a *hierarchy of abstraction* is constructed (see figure 4). The abstraction mechanism to arrange information in a human-like manner improves, among others, human-robot communication, as demonstrated in [9], [10], [13].

Within the World Model module different skill units are devoted to create and manipulate world symbolic information. In our current implementation there is a human skill unit that permits the wheelchair user to create spatial symbols like distinctive places or rooms in a hierarchical graph-based structure. To do that she/he must manually guide the vehicle to a certain location where geometrical information, like feature segment maps and pose information, is automatically added to the internal model. Topological information is created by the user through the commands (0)-(3) of figure 5, establishing verbally linguistic labels for topological elements, i.e. ”this is a distinctive place called Peter’s office”. Please refer to [9] for a further explanation on this human assisted world model creation.

### *B. Task Manager*

The Task Manager module receives requests of tasks to be planned and carried out by the robot. In our current implementation only the user can request robot tasks, but in general it could be useful to accept requests from other agents: for example from a supervisor if the wheelchair serves an elderly person at a geriatric, or from software applications, i.e a system procedure that checks the battery level of the robot could request a battery-recharge task.

Skill units of the Task Manager module attend requests from those different ”clients”[4]: robotic units should permit applications, or even other machines to request the robot for some operation, while human skill units enable people (in our case only the wheelchair user) to ask the robot for task execution via a voice interface.

### *C. Task Planner*

Considering the robotic (and human) physical abilities, the Task Planner module generates plans (i.e., a sequence of robot+human basic actions) to attain the goals requested by the Task Manager. It takes into account the spatial information stored in the World Model module and the available set of human and robot abilities from a previously defined planning domain (see figure 6). It is relevant to notice that the Task Planner module does not only produce

---

[4] In case of multiple clients requesting tasks to the mobile robot, the processing core of the Task Manager CMS should implement a priority policy between its different skills units.

| Id | Human verbal commands | Robot responses |
|---|---|---|
| (0) | This is a distinctive place called <free string> | Ok, place <free string> added |
| (1) | Group previous locations in a room called <free string> | Ok, room <free string> added |
| (2) | Open door between <place1> and <place2> | Ok |
| (3) | Closed door between <place1> and <place2> | Ok |
| (4) | No alerts | Setting normal state |
| (5) | Low collision risk | Setting low level alert state |
| (6) | High collision risk | Setting high level alert state |
| (7) | Stop | Ok, stopping |
| (8) | Cancel | Ok, cancelling action |
| (9) | Continue | Ok, continuing the navigation |
| (10) | Take me to <distinctive place> | Ok // That place does not exist |
| (11) | I want to guide you | Ok |
| (12) | Select another method | Ok // No available methods, May you help me? |

|  | Robot commands to the human | Accepted human responses |
|---|---|---|
| (13) | Please can you guide us to <distinctive place> | Yes // No, I can not |
| (14) | Please can you open the door which is in front of you | yes // No, I can not |

|  | Human acknowledgment information | Accepted human responses |
|---|---|---|
| (15) | I have guided you to <distinctive place> | Thank you |
| (16) | I could not reach to <distinctive place> | I can not continue, please select another destination |
| (17) | I have just opened the door | Thank you |
| (18) | I could not open the door | I can not continue, please inquire external help |

Fig. 5. Human-Robot verbal communication. This table describes the verbally interaction between the user and SENA which has been considered in our experiences. To improve communication we have extended the grammar to accept small variations, i.e. ”Let’s go to <distinctive place>” is recognized as a variant of the command (10)

a sequence of actions, but it may also suggest the most efficient method (the skill unit from the functional layer) to perform each one. The planning algorithm used in our work, *Metric-FF* [20], is able to produce an optimized plan with respect to a certain criteria. In our implementation, plans are optimized with respect to hand-coded cost functions which yield approximations of the execution cost of skill units. For example, function (1) yields an approximate cost of a reactive navigation.

$$Cost(Reactive\#1, l1, l2) = k_1 * distance(l1, l2) \tag{1}$$

where $l1$ and $l2$ are the origin and destination locations respectively, and $k_1$ is a constant value that measures a certain metric like the time spent by the navigational algorithm to travel each distance unit. The cost of human actions is fixed to a high value to avoid the planning process to select human abilities whereas there are alternative robotic ones.

```
(define (domain navigation)
(:requirements :typing :fluents)
(:types location object)
(:const Robot object)
(:predicates
                    (at ?obj - object ?loc - location)
                    (link ?x ?y - location)
                    (open ?obj -object)
                    (closed ?obj -object))
(:functions
                    (cost-reactive ?l1 ?l2 - location)
                    (cost-manually-guided ?l1 ?l2 - location)
                    (navigation-cost))
(:action OPEN-MANUALLY-DOOR
          :parameters
          (loc-from - location
          ?door - object)
:precondition
          (and (at Robot ?loc-from)
          (closed ?object))
          (increase navigation-cost 10)
:effect
          (and (not (closed ?object)) (open ?obj)))
(:action MOVE-REACTIVE
:parameters
          (loc-from - location
          ?loc-to - location)
:precondition
          (and (at Robot ?loc-from)
          (link ?loc-from ?loc-to))
:effect
          (and (not (at Robot ?loc-from)) (at Robot ?loc-to)
          (increase navigation-cost (cost-reactive ?loc-from ?loc-to))))
(:action MOVE-MANUALLY-GUIDED
:parameters
          (loc-from - location
          ?loc-to - location)
:precondition
          (and (at Robot ?loc-from)
          (link ?loc-from ?loc-to))
:effect
          (and (not (at Robot ?loc-from)) (at Robot ?loc-to)
          (increase navigation-cost (cost-manually-guided ?loc-from ?loc-to)))))
```

Fig. 6. Example of planning domain. It includes three human-robot abilities, two navigation methods performed by a robotic and a human skill unit respectively, and a manipulation method (open-door) also performed by the human. The parameter `navigation-cost` yields the planning cost involved in navigation tasks.

Thus, through the planning domain described in figure 6 and the initial situation depicted in figure 4 (considering the door $D3$ closed and all the rest opened), the resultant plan that solves the task *"Go to R2-2"* could be: `MOVE-REACTIVE (H2,WC3), OPEN-MANUALLY-DOOR (D3), MOVE-REACTIVE (D3, R2-2)`, in which the human help is only required to open the closed door.

Human integration at the deliberative level also enables her/him to interactively participate in the planning process. As presented elsewhere [13], [14], the planning process can use the hierarchical arrangement of the internal model to produce plans at a certain intermediate level of abstraction, consulting the user whether such scheme for a plan (an abstract plan) meets her/his wishes. This feature permits humans to accept or reject the proposed plan inquiring a different solution or providing a new one to attain the goal. For example, a possible plan that solves the task "Go to Laboratory" in the situation depicted in figure 4 could be: "Navigate (Office1, West Corridor), Navigate (West Corridor, Laboratory)". Human integration in the planning process enables her/him to, for example, reject such an abstract plan if one of its actions involve an abstract concept, i.e. "West Corridor" not desired by the human, inquiring an alternative solution (in this case, "Navigate (Office1, East Corridor), Navigate (East Corridor, Laboratory)”).

## V. EXECUTIVE INTEGRATION

The Execution and Control layer works as an intelligent bridge between the deliberative and the functional layer. It asks modules of the functional layer for the execution of basic actions. During the execution of actions, it also processes the information gathered from sensors to check and avoid risky situations, such as collisions, power failures, etc., deciding the most convenient robot reaction, for instance, stopping the vehicle and waiting for human help.

Human integration plays a relevant role in this layer, since she/he can provide her/his perceptual ability (goal (b) in the introduction) to report danger situations, i.e. ”there is a collision risk”, as well as to decide what to do when an anomalous circumstance occurs, i.e. ”I want to take the motion control” (please, refer again to table of figure 5 for all available human commands). These features provided by ACHRIN are of special significance for improving robot dependability.

The execution and control layer is composed of two modules: the *Alert System* (section V-A) module which registers external stimuli and the *PLEXAM* module (section V-B) that sequences and controls plans’ executions.

### *A. Alert System*

The Alert System module checks for unexpected dangerous situations through its skill units. Robotic skill units can warn the robot about, for example, collisions or low-battery situations through the reading of the robot sensors. In addition, the wheelchair user is integrated into the Alert System through human skill units which incorporate the capability to inform about risky situations using verbal commands like ”there is a collision risk” or ”the battery level is low”.

The integration of the human into the alert system of ACHRIN improves the reliable performance of the robot, since she/he can report danger situations not detected by robot sensors. Besides, the human can also predict future risky situation based on her/his knowledge about the environment, and thus, for example, she/he can notice about a collision risk when the vehicle is close to enter a room which is probably crowded.

The Alert System may distinguish different risk levels to adequate the behavior of the vehicle operation (see figure 7). Thus, for the case of collision alerts[5] we have defined the following four alert levels based on both human indications and fix thresholds for readings of the laser scanner of our robot[6]:

- *Normal*. No obstacles have been detected, and thus the system works without any danger.
- *Minimal Risk*. An obstacle has been detected at a safety distance, and thus a minimal risk is considered. This low-level risk may produce variations on the execution parameters of the current robot action, like reducing the vehicle velocity.

---

[5] A similar alert categorization can be done for other situations like low-level battery charge.

[6] In our current implementation the sensor-based alerts are triggered based on fixed thresholds, but a more flexible mechanism can be implemented to adapt the vehicle behaviour to the human preferences, learning such a threshold from experience [2]).

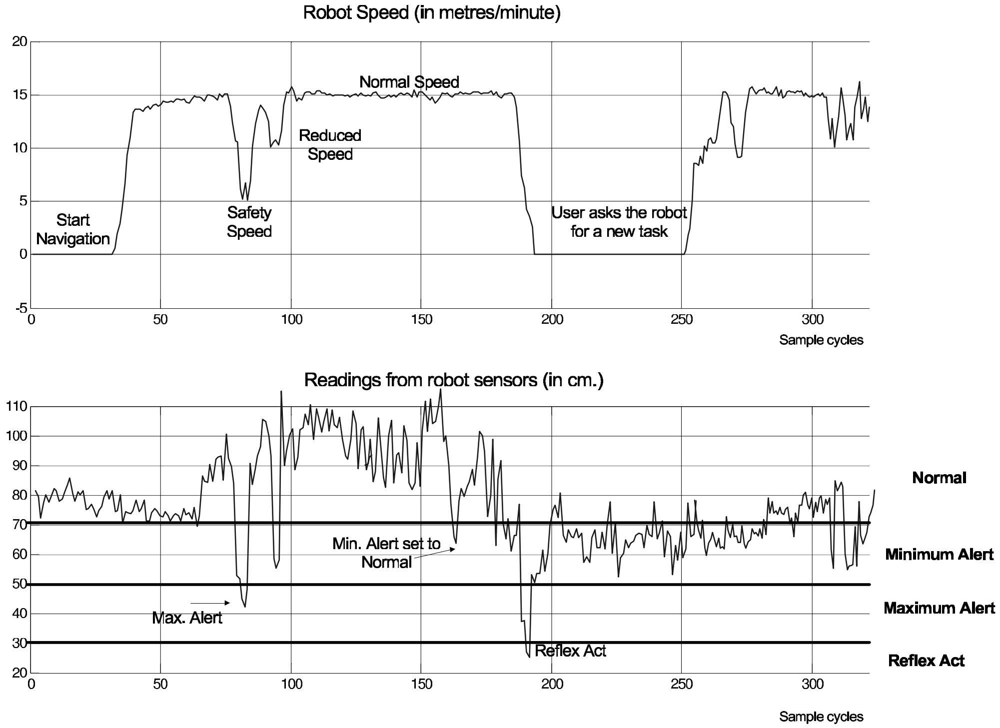


Fig. 7. Evolution of collision alert levels over time in a real experiment. Top: Speed of the robot during part of our experiments (1 sample cycle=0.5 sec.). Bottom: Minimum distance measured by robot sensors to the closest obstacle. Alerts level threshold is marked with a thick line. Notice that the user can ignore, for example, a minimum alert by setting the alert level to a normal state (around cycle 160).

- *Maximum Risk*. A critical situation has been reported by a robotic or human skill unit. In this case, the module that executes the current robot action must modify its parameters to prevent any danger, i.e. reducing drastically the vehicle speed.
- *Reflex Act*. A reflex act is the most critical level. When an obstacle is detected too close, a reflex act is triggered, stopping the vehicle to avoid any possible damage. The user must recover this situation by requesting a new task.

The user plays a dominant role in the Alert System module since alerts reported by her/him are prioritized. Thus, the user can ignore, for instance, a minimal risk from a collision skill unit that uses the readings of a laser rangefinder sensor, setting the alert level to normal[7] (see figure 7). This feature permits humans to modulate the behavior of the robot to her/his wills, i.e. when the user, intentionally, wants to closely approach a wall.

Chart of figure 7 shows the evolution of the collision alert levels in part of our experiments in which three different situations can be observed: 1) A maximum alert is detected near cycle 80, decreasing the robot speed to a

---

[7] When the user ignores alerts, i.e. collision, the system does not consider information from sensors which can provide that alert during a certain period of time (set in our implementation to 5 secs.), considering then only human alerts. After that period of time, the system automatically considers again any possible alert from any source.

safety value (5 m/min). 2) Minimal risk alerts (around cycle 160) are ignored by the user who establishes a normal state (vehicle speed to 15 m/min). 3) Finally, a collision reflex (before cycle 200) is detected causing the detention of the vehicle and waiting for a new task.

Other robot architectures also include alert mechanisms (see for example [5], [8]) but they do not incorporate the user capability for consciously ignoring certain alerts from the environment, which is of much interest in clutter scenarios.

### B. Plan Executor and Alert Manager (PLEXAM)

This module sequences plans' actions and controls their execution based on the information gathered by the alert system. During the execution of a particular action, PLEXAM registers system alerts according to their severity and communicates them (if any) to the functional layer in order to accordingly tune their performance (see figures 7 and 9).

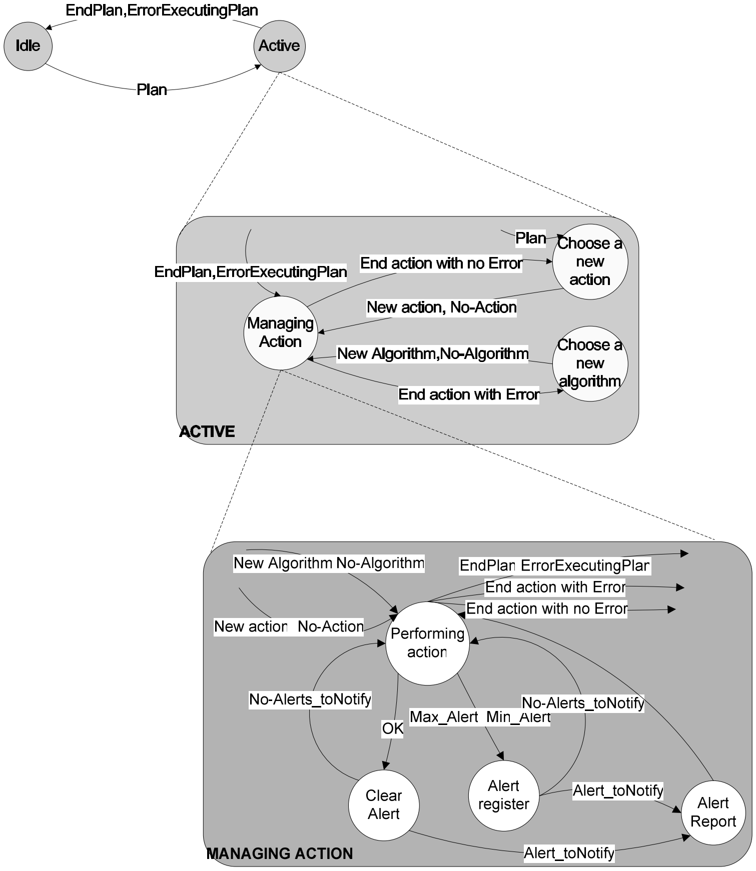


Fig. 8. PLEXAM state graph. This module sequences actions while maintaining a list of all alerts registered in the system. It notifies alerts to functional modules at the same time that it waits for their results. Based on such results, PLEXAM selects a new action or a new algorithm (skill unit to perform the next action).

PLEXAM is also in charge of detecting malfunctions of skill units. For instance, a failure in the execution of a navigation skill is considered when it has not responded (reporting neither success nor failure) after a certain period of time. In that cases, PLEXAM is responsible for cancelling the current execution and deciding the next skill unit to finish the action.

The human is also integrated in this module by a human skill unit that enables the user to report execution failures (goal (c) in the introduction) deciding the selection of another skill unit to finish the action given the current environmental conditions. In the case of navigation failures, for example, the user can decide to take the control of the vehicle through the manually-guiding method. Such a participation of the human in the robot navigation is restricted, as any other functional component of ACHRIN, by the safety issues imposed by the Alert System. Thus, when the user guides the vehicle, PLEXAM can stop it due to a collision alert, ignoring the user motion inputs and choosing the next available skill unit to accomplish the action. In the case of failure of all available skill units (including the human ones), PLEXAM cancels the execution of the current plan, inquiring the user for a new achievable goal.

In the absence of failures, when the functional layer reports the successfully execution of the requested action, PLEXAM continues the execution of the next action (if any) of the plan. Figure 8 details the PLEXAM work through its state graph.

## VI. FUNCTIONAL INTEGRATION

The Functional (and lowest) layer of the architecture contains a set of modules that physically materialize the abilities of the human-robot system. In the general case of mobile robots, this layer must contain at least a functional module to perform navigation; however, additional modules can be also considered to perform other kind of actions such as manipulation, inspection, social interaction, etc. Besides, each module can entail a number of skill units to perform an action, i.e. navigation, in different ways (reactive, tracking a computed path, or manually guided).

**Speed parameter (metres/minute)**

| **Navigation skill Unit** | **Normal** | **Minimal Risk** | **Maximum Risk** |
|---|---|---|---|
| Manually guided | 12 | 10 | 5 |
| Reactive | 15 | 10 | 5 |
| Tracked | 10 (from 9:00 to 12:00am)<br>20 (the rest of the day) | 8 | 4 |

Fig. 9. Example of the variation of the speed parameter for navigation skill units under alerts occurrence.

In this layer, the human is integrated into the robotic system augmenting and/or improving the robot capabilities (goal (a) in the introduction), i.e. an user can help the vehicle to manipulate objects (open a door) or she/he can recover the robot from a navigation failure, manually guiding the vehicle as shown in section VII.

The processing core of functional modules takes care of alerts reported by PLEXAM to establish the execution parameters of skill units, i.e. the maximum speed of a reactive algorithm in a normal situation (no alerts are reported) is set to $15$ metres/minutes and to $10$ when a minimal risk is reported (see figure 7). Such values can be fix and hand-coded for every skill unit or variable depending, for instance, on the current time (see table of figure 9).

The execution of skill (human or robotic) units from the functional layer may finish due to three different causes:

- The unit successfully executes its action. This is the normal ending status of the skill units which is communicated to PLEXAM to launch the next action (if any) of the plan.
- The execution is cancelled by PLEXAM. It can cancel the execution of a skill because of a detected failure (an example of this circumstance is described in VII).
- The skill unit execution finishes without achieving its goal due to an unforeseen circumstance or a failure.

## VII. HUMAN-ROBOT INTEGRATION IN A ROBOTIC WHEELCHAIR

The demonstration of the suitability of a robotic architecture to a particular application is not easily quantifiable. In our work we have tested ACHRIN by performing many real experiences on SENA, within a real environment (our research building) involving six different-ages individuals (five male and one female) who have confirmed their satisfaction while using the vehicle (see figure 10).

Fig. 10. Experiments with the SENA robotic wheelchair. These images are snapshots of videos (some of them appeared in life tv shows), which can be found at [36].

The robotic ability to navigate between two distinctive places (given by the task-planner) relies on a reactive navigation using the laser rangefinder and the ring of infrared sensors [3]. From the human part, we allow the user to improve the navigation of the robot by manually guiding the vehicle and to provide manipulation abilities to open doors. She/he can also notice the system about collision alerts as well as system failures.

For a natural human-robot interaction, commercial voice recognition/generation software [41], [42] has been used. The set of considered commands and their possible responses, enumerated in figure 5 (in the following, commands' IDs will be used to describe our experiments), are specified in a grammar file, used by the voice recognizer software. Such a grammar, given in a BNF format, can be easily extended to cope with additional commands.

Although current voice recognizer software exhibits a high level of accuracy, it sometimes does not work properly, mainly due to a noisy environment or a poor training by the user. In our application non-recognized utterances are detected (they do not fit the grammar), asking the user for repetition, i.e. ”Pardon?”, or ”Can you repeat, please?”. Non-recognition of critical human commands, like ”Stop”, will not suppose any risk of crashing due to the concurrent work of the Alert System.

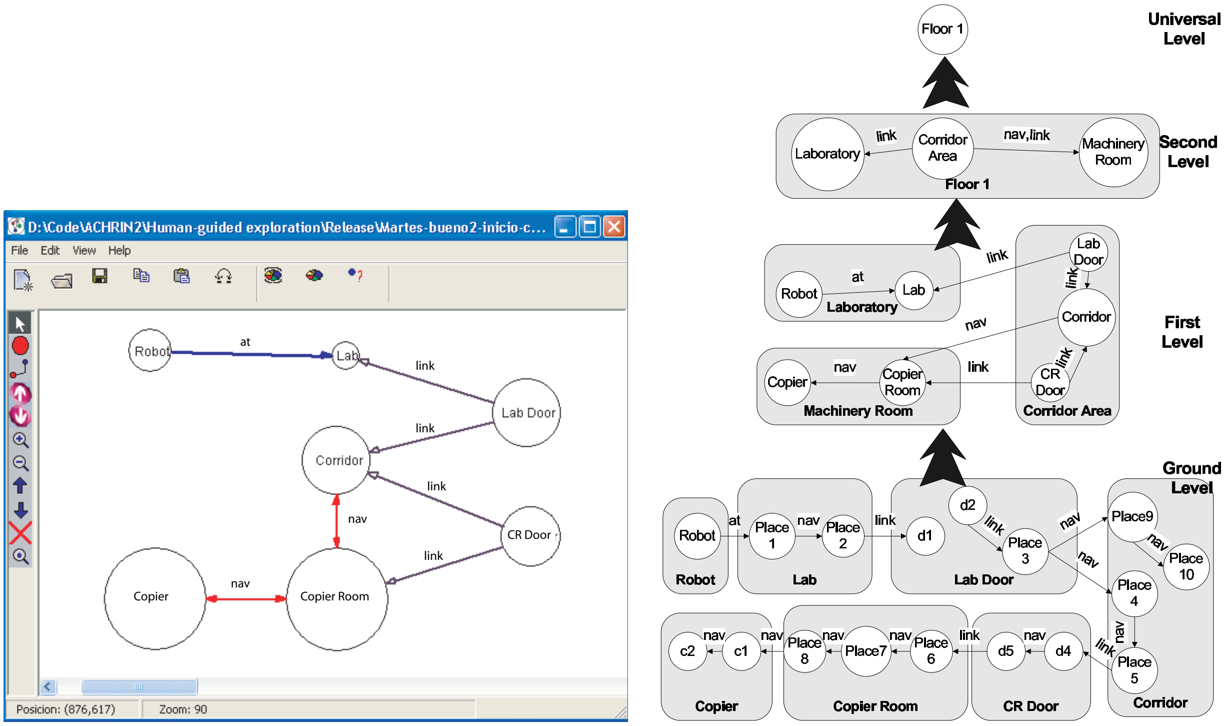
In our tests, users were informed about the available grammar and the set of possible destinations. They also tested the world model creation by using free-context utterances to name new distinctive places. Table 11 shows the speech recognition results in our experiences. As expected, the percentage of recognized commands is clearly higher than recognition of free-context strings. When wrong labelling of distinctive places occurs, the user can correct it through a GUI application (see figure 12a).

In a high percentage of cases, users reported a failure of the robotic navigation because of a low performance of the reactive algorithm to escape from tight spaces, as commented further on. Since our experiments include a number of passing through doors and narrow passages, the user often alerted the system about collisions though there were no risk.

| **Speech Recognition Results** | |
|---|---|
| Recognized commands | 83.4% |
| Wrong recognized commands | 1.3% |
| Recognized free-context utterances | 12.4% |
| **Navigation Results** | |
| Robotic navigation failures | 23.4 % |
| Human detection of a robotic failure | 45.2 % |
| Human navigation failure | 8.7 % |
| **Alert System Results** | |
| Alerts ignored by the user | 3.4 % |
| Alerts produced by the user | 27.8 % |
| Reflex alerts | 7.7 % |

Fig. 11. Results from our tests. Percentages are calculated based on five different navigation experiments performed by six different people (that is, 30 experiences).

Next we detail a particular navigational experience. The first part of our tests consists of endowing SENA with a symbolic model of the workspace used to plan and execute tasks (see [9] for a detailed description). This model is created by the wheelchair user through a human skill unit (within the World Model Module) that responds to commands (0)-(3) to create topological information. Geometrical information (like geometrical maps and robot pose) needed for posterior navigation is automatically attached to nodes and arcs that represent topological locations in the model. Figure 12 depicts an abstract level of the internal model managed by SENA as well as the spatial hierarchy used in our experiences.



a) b)

Fig. 12. Internal World Model. a) The internal world model with linguistic labels is shown through a graphical interface (only the first level). Arcs indicate relations between spatial symbols, like navigability. The state of doors (opened or closed) is deduced from node connectivity, i.e., ”Lab door” is closed since there are not navigability arcs between ”Lab” and ”Corridor”. b) Spatial hierarchy considered in our experiments. Ground and first level are created by the human-robot system, while the others have been include for completeness.

| **Plan P** |
|---|
| MOVE-REACTIVE *Lab, Lab-door* |
| OPEN-MANUALLY-DOOR *Lab-door* |
| MOVE-REACTIVE *Lab-door, Corridor* |
| MOVE-REACTIVE *Corridor, Copier-Room-door* |
| MOVE-REACTIVE *Copier-Room-door, Copier* |

Fig. 13. Sequence of skills to go from *Lab* to the *Copier*.

Once a model of the environment is available, the user can select a destination through a human skill unit within the Task Manager module that attends to command (10), for instance ”Take me to the *Copier*”. Using the information stored in the World Model module (figure 12) and the planning domain of figure 6, the resultant plan, *P*, yielded by the Task Planner module is shown in figure 13.

In the generation of plan *P*, human help is only considered when there is not an available robot ability to achieve the goal, i.e., open a door. During the plan execution (see figure 15 a), in absence of failures on the robotic unit

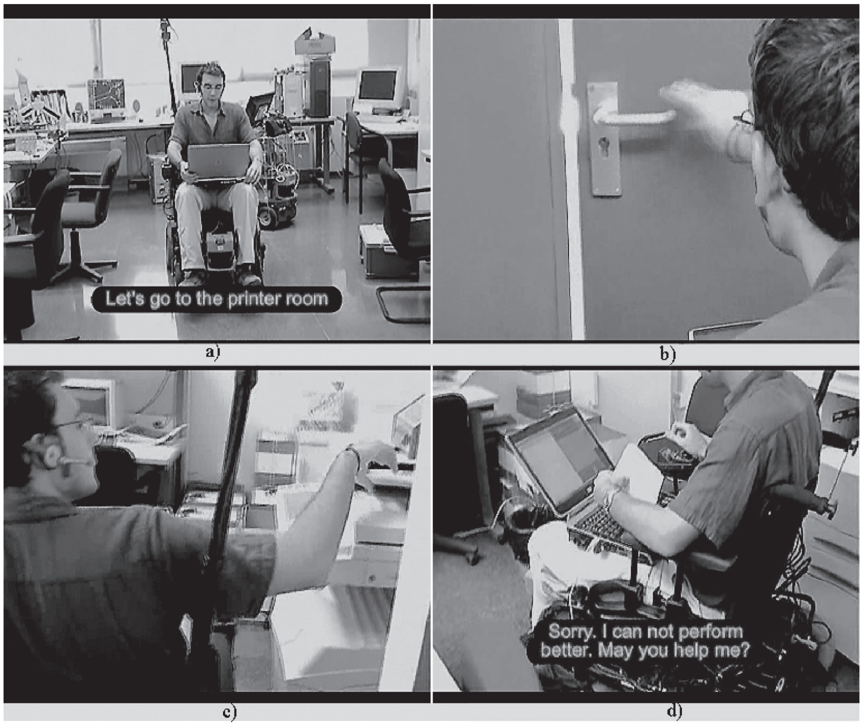


Fig. 14. Execution of navigation tasks. a)The human asks the robot for the execution of a task. b)The robotic system requires the human help to open a door. c)The vehicle reaches the destination: the copier-room. d) The user recover the robot after a navigation failure.

that performs navigation, PLEXAM only inquires the human help (through command (14)) after the success of the first action (MOVE-REACTIVE *Lab, Lab-door*). When the user reports the success of the entrusted action (human acknowledgement (17))[8], the execution of *P* continues autonomously until the destination is achieved. Figure 14 shows some pictures taken during the execution of plan P.

Our experiences assume that the information stored in the internal model is coherent with the real environment and that there are not external agents modifying the robot workspace. Thus, during planning no attention has been paid to model incoherences like opening a door which is just opened or traversing a closed door. However, failures produced for such inconsistencies are correctly managed by human integration. In the latter situation, for example, the integrity of the system is ensured by the alert system which would report a collision alert stopping the vehicle while the user should modify the internal model of the system through command (3).

In other experiences, we have also tested human-robot integration at the intermediate level of ACHRIN to control the execution of plans. Thus, after the execution of plan *P*, the user asked SENA to go back to the laboratory

---

[8] Note that opening a door while sitting on a wheeled chair may turn into an arduous task. In some cases in which the user could not open the door she/he could ask surrounded people for help.

a) b)

Fig. 15. Occupancy gridmaps from our test scenario. a) The route marked in the map corresponds to the navigation of the vehicle during the execution of the plan *P*. b) Path followed during the execution of $P^*$. The thicker section of the path corresponds to the manually guidance.

(command (10)). In this case the resultant plan $P^*$ is similar to *P* but in a reverse order and without the ”open door” action, since it is supposed to be opened after the execution of plan *P*. Plan $P^*$ only contains navigational actions, thus the Task Planner module does not consider human participation to solve the task. The execution of $P^*$ starts normally with the PLEXAM call to the robotic skill unit that performs the action MOVE-REACTIVE (*Copier*, *Copier-Room-door*). In the test scenario, our reactive algorithm usually fails when performing this action due to the lack of space for manoeuvering (see figure 15 b). In this case, users normally reports a navigation malfunction of the robotic unit through a human skill unit within PLEXAM, cancelling the vehicle navigation (command (8)) and inquiring the selection of an alternative method (command (12)). The human help is required by PLEXAM via command (13), ”Can you guide us to *Copier-Room-door*?”. Once the user concludes the action and reports the success of the manually guidance of the vehicle to the destination, the execution of the plan is resumed. Figure 17 shows the flow of information between modules during the execution of plan $P^*$.

Notice that when the human help is required to accomplish an action she/he can refuse the order or may not properly finished it, for example, she/he may not be able to open a closed door. In that cases, since human help is only required as a last resort, the system stops, waiting for external help (from a surrounded person) or another achievable goal commanded by the user.

System collision alerts were also tested in our real experiences. During the navigation of the wheelchair, i.e. executing plans like *P*, the alert system continuously checks for collision risks from two sources: the rangefinder sensors (laser scanner and infrared ring) mounted on SENA and the alert messages reported by the user (through commands (5)-(7)). Figure 16-top shows the distance yielded by the rangefinder sensors to the closest obstacle during the navigation from point *A* to *C* (see figure 15. Based on this information and the defined alert thresholds (0.2 m. for reflex, 0.5 m. for maximum alert, and 0.75 m. for minimum alert), figure 16-middle shows the evolution

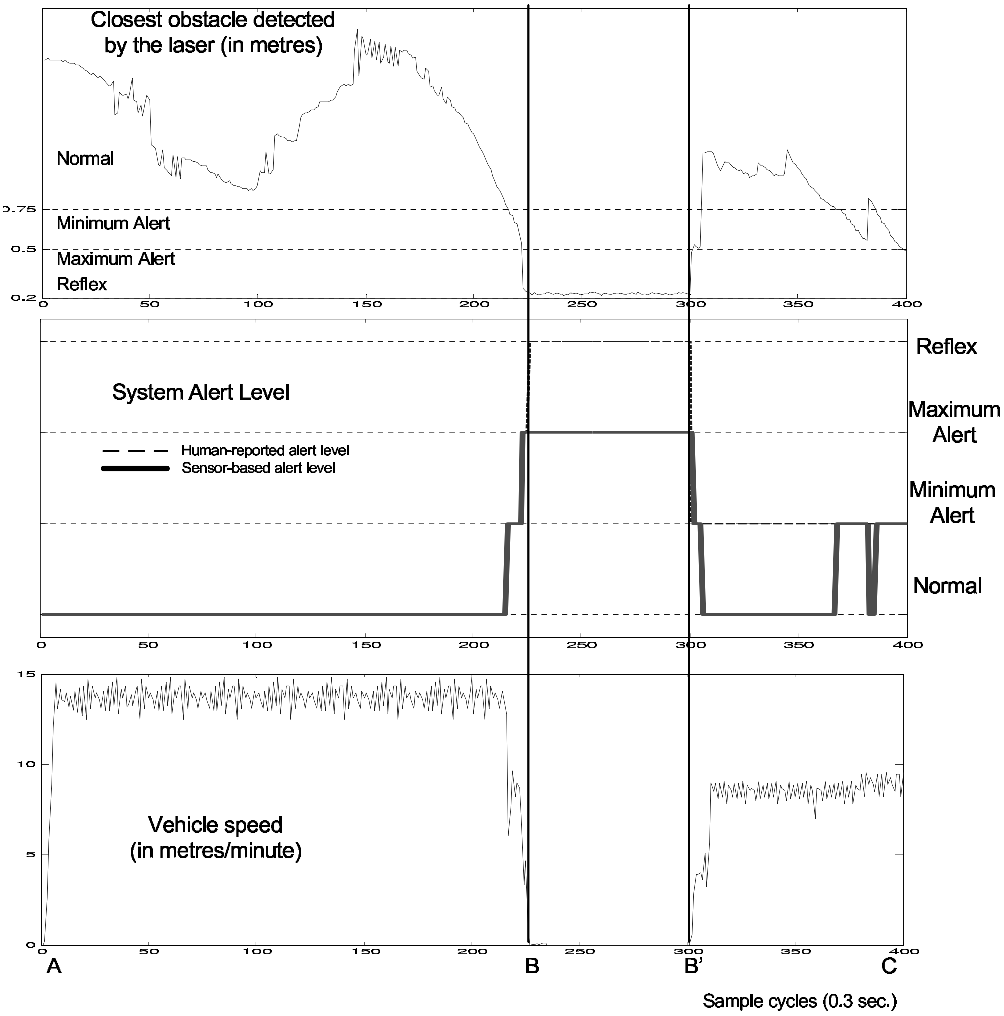


Fig. 16. Collision alerts test. Top) Distance to the closest obstacle during part of the wheelchair navigation measured by the laser rangefinder. Middle) System alert level along the experience. Notice how the user can set a certain alert level more restrictive than the sensor-based alert. Bottom) The vehicle speed during the experience. Speeds depends on the system alert level (15, 10, and 5 m/min. for normal, minimum and maximum levels, respectively). In the interval *B-B'*, in which the user triggered a reflex act, the speed is set to 0.

of the system alert level during a section of an experiment along a clutter environment. In this figure we distinguish the alert source: human or sensor-based alert. Human alerts prevail over sensor-based ones, and thus, in this example, between points *B* and *B'* (around cycles $225$-$300$) the user sets the alert level to *Reflex* near the open door, albeit the rangefinder readings report a *Maximum alert*. In the same manner, from point *B'* to the destination (point *C*) the alert level is set to *Minimum alert* by the user in spite of the rangefinder information. Finally, figure (16-bottom) shows the robot speed during the navigation and its adaptation to the current alert level.

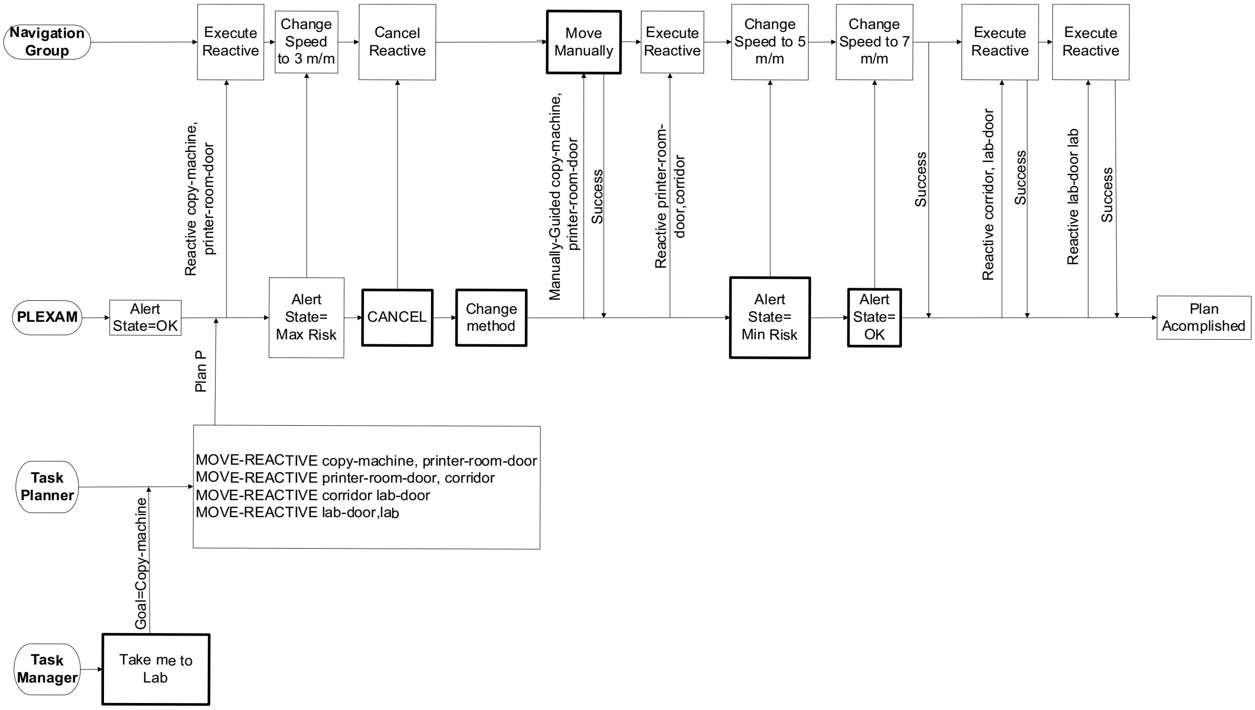


Fig. 17. Scheme of some modules' communication during the execution of a plan. The flow of information is from bottom to top and from left to right. Thick boxes represents human skill units.

## VIII. CONCLUSIONS AND FUTURE WORK

This paper has proposed the integration of humans into the robotic system as the best way to extend/augment mobile robot performance in real scenarios. Such an integration is significant in certain robotic applications in which human and robot are closely related. That is the case of assistant robotic applications.

The proposed human-robot integration has been materialized through a control architecture called *ACHRIN*. ACHRIN has been devised to enable humans at all levels, ranging from performing low-level navigation like manoeuvering in complex situations, to high-level decision-makings, like symbolically model the environment.

The proposed architecture has been implemented and largely tested on a real assistant robot, the robotic wheelchair *SENA*, illustrating its suitability to this kind of applications.

## REFERENCES

[1] P. Aigner and B. McCarragher. Human integration into robot control utilising potential fields. In *IEEE Int. Conf. on Robotics and Automation, Albuquerque, NM*, pages 291–296, 1997.

[2] J.L. Blanco, A. Cruz, Galindo, Fernández-Madrigal, and J. González. Towards a multi-agent software architecture for human-robot integration. In *4th International Workshop on Practical Applications of Agents and Multiagent Systems (IWPAAMS), León (Spain)*, pages 235–244, 2005.

[3] J.L. Blanco, J. González, and J.A. Fernández-Madrigal. Consistent observation grouping for generating metric-topological maps that improves robot localization. In *IEEE Int. Conf. on Robotics and Automation, Orlando*, May 2006.

[4] J.W. Crandall and M.A. Goodrich. Experiments in adjustable autonomy. In *IEEE Int. Conf. on Systems, Man, and Cybernetcis, Tucson, AZ*, pages 1624–1629, 2001.

[5] R.I. Damper, R.L.B. French, and T.W. Scutt. Arbib: An autonomous robot based on inspirations from biology. *Robotics and Autonomous Systems*, 31:247–274, 2000.

[6] G. Dorais, R.P. Bonasso, D. Kortenkamp, B. Pell, and D. Schreckenghost. Adjustable autonomy for human-centered autonomous systems on mars. In *Proc. of the First International Conference of the Mars Society*, pages 397–420, 1998.

[7] R. Falcone and C. Castelfranchi. The human in the loop of a delegated agent: the theory of adjustable social autonomy. *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, 31(5):406–418, 2001.

[8] J.L. Fernandez and R.G. Simmons. Robust execution monitoring for navigation plans. pages 551 – 557. IEEE/RSJ IROS 1998, Victoria B.C and Canada, 1998.

[9] J.A. Fernández-Madrigal, C. Galindo, and J. González. Assistive navigation of a robotic wheelchair using a multihierarchical model of the environment. *Integrated Computer-Aided Engineering*, 11:309–322, 2004.

[10] J.A. Fernández-Madrigal and J. González. *Multi-Hierarchical Representation of Large-Scale Space. International Series on Microprocessor-based and Intelligent Systems Engineering. Vol. 24*. Kluwer Academic Publishers, Netherlands, 2001.

[11] T. W. Fong, C. Thorpe, and C. Baur. Collaboration, dialogue, and human-robot interaction. In *10th International Symposium of Robotics Research, Lorne, Victoria, Australia*, pages 255 – 270, London, November 2001. Springer-Verlag.

[12] T.W. Fong and C. Thorpe. Robot as partner: Vehicle teleoperation with collaborative control. In *In Proceedings of the NRL Workshop on Multi-Robot Systems*, pages 195–202, 2002.

[13] C. Galindo, J. González, and J.A. Fernández-Madrigal. Assistant robot, interactive task planning through multiple abstraction. In *16th European Conference on Artificial Intelligece, Valencia, Spain*, pages 1015–1016, 2004.

[14] C. Galindo, J. González, and J.A. Fernández-Madrigal. Interactive task planning in assistant robotics. 5th Symposium on Intelligent Autonomous Vehicles, Lisbon, 2004.

[15] J. Goetz and S. Kiesler. Cooperation with a robotic assistant. In *CHI '02: Extended abstracts on Human factors in computing systems*, pages 578–579, New York, NY, USA, 2002. ACM Press.

[16] S. Harnand. *Psychological and Cognitive Aspects of Categorical Perception: A Critical Overview*. Harnan S. (ed.), New York, Cambridge University Press, Chapter 1, 1987.

[17] S.C. Hirtle and J. Jonides. Evidence of hierarchies in cognitive maps. *Memory and Cognition*, 13(3):208–217, 1985.

[18] C.S. Holding. Further evidence for the hierarchical representation of spatial information. *Journal of Environmental Psychology*, 14:137–147, 1994.

[19] F.F. Ingrand and E. Py. An execution control system for autonomous robots. In *IEEE Int. Conf. on Robotics and Automation, Washington, DC*, pages 1333–1338, 2002.

[20] H. Jörg and N. Bernhard. The ff planning system: Fast plan generation through heuristic search. *J. of Artificial Intelligence Research*, 14:253–302, 2001.

[21] K. Kawamura, R.A. Peters, C. Johnson, P. Nilas, and S. Thongchai. Supervisory control of mobile robots using sensory egosphere. In *IEEE Int. Symposium on Computational Intelligence in Robotics and Automation, Banff, Alberta*, pages 531–537, 2001.

[22] O. Khatib. Human-centered robotics and haptic interaction: From assistance to surgery, the emerging applications. In *Third International Workshop on Robot Motion and Control*, pages 137–139, 2002.

[23] M. Kleinehagenbrock, J. Fristch, and G. Sagerer. Supporting advanced interaction capabilities on a mobile robot with a flexible control system. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Sendai, Japan*, pages 3649–3655, 2004.

[24] S. Kristensen, J. Klandt, F. Lohnert, and A. Stopp. Human-friendly interaction for learning and cooperation. In *IEEE Int. Conf. on Robotics and Automation, Seoul, Korea*, pages 2590–2595, 2001.

[25] B.J. Kuipers. *The Cognitive Map: Could it Have Been Any Other Way?. Spatial Orientation: Theory, Research, and Applications*. Picks H.L. and Acredolo L.P and New York, Plenum Press, 1983.

[26] B.J. Kuipers. The spatial semantic hierarchy. *Artificial Intelligence*, 119:191–233, 2000.

[27] S. Levine, D. Bell, L. Jaros, R. Simpson, Y. Koren, and J. Borenstein. The navchair assistive wheelchair navigation system. *IEEE Transactions on Rehabilitation Engineering*, 7(4):443–451, 1999.

[28] M. Lindstrom, A. Oreback, and H.I. Christensen. Berra: A research architecture for service robots. In *IEEE Int. Conf. on Robotics and Automation, San Francisco, CA*, pages 3278–3283, 2000.

[29] X. Lv and X. Huang. Three-layered control architecture for microassembly with human-robot task plan interaction. In *IEEE Int. Conf. on Robotics and Biomimetics, Shenyang, China*, pages 617–622, 2004.

[30] H.F. Machiel, J.J. Wagner, N. Smaby, K. Chang, O. Madrigal, L.J: Leifer, and O. Khatib. Provar assistive robot system architecture. In *ICRA*, pages 741–746, 1999.

[31] K. Morioka, J.H. Lee, and H. Hashimoto. Human centered robotics in intelligent space. In *In Proceedings of the IEEE International Conference on Robotics and Automation, Washington, DC*, May 2002.

[32] S.P. Pari, R.S. Rao, S. Jungt, V. Kumar, J.P. Ostrowski, and C.J. Taylor. Human robot interaction and usability studies for a smart wheelchair. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Las Vegas, Nevada*, pages 3206–3211, 2003.

[33] M. Pollack, S. Engberg, J.T. Matthews, S. Thrun, L. Brown, D. Colbry, C. Orosz, S. Ramakrishnan B. Peintner, J. Dunbar-Jacob, C. McCarthy, M. Montemerlo, J. Pineau, and N. Roy. Pearl: A mobile robotic assistant for the elderly. In *Workshop on Automation as Caregiver: the Role of Intelligent Technology in Elder Care (AAAI)*, August 2002.

[34] R.S. Rao, K. Conn, S.H. Jung, J. Katupitiya, T. Kientz, V. Kumar, J. Ostrowski, S. Patel, and C.J. Taylor. Human robot interaction: Application to smart wheelchairs. In *IEEE Int. Conf. on Robotics and Automation, Washington, DC*, pages 3583–3588, 2002.

[35] E. Remolina and B.J. Kuipers. Towards a general theory of topological maps. *Artificial Intelligence*, 152:47–104, 2004.

[36] SENA robotic wheelchair. Homepage http://www.isa.uma.es/ent_inv.htm.

[37] P. Scerri, D. Pynadath, and M. Tambe. Adjustable autonomy in real-world multi-agent environments. In *AGENTS '01: Proceedings of the fifth international conference on Autonomous agents*, pages 300–307, 2001.

[38] J. Scholtz. Theory and evaluation of human-robot interaction. pages 1–10. 36th International Conference on System Sciences, Hawai, 2003.

[39] T.B. Sheridan. *Telerobotics, Automation and Human Supervisory Control*. MIT Press,Cambridge, MA, 1992.

[40] K.A. Tahboub. A semi-autonomous reactive control architecture. *J. Intell. Robotics Syst.*, 32(4):445–459, 2001.

[41] Text to Speech Software 2nd Speech Center. http://www.zero2000.com.

[42] IBM ViaVoice. http://www-3.ibm.com/software/voice/viavoice/.

[43] G. Wasson, J. Gunderson, S. Graves, and R. Felder. An assistive robotic agent for pedestrian mobility. In *AGENTS '01: Proceedings of the fifth international conference on Autonomous agents*, pages 169–173, 2001.