To appear in Applied Bionics & Bionmechanics, Aug. 2006

A Multi-Agent Control Architecture for a Robotic Wheelchair

Galindo C., Cruz-Martin A., Blanco J.L., Fernández-Madrigal J.A., and Gonzalez J.

E.T.S.I. Informática – Universidad de Málaga

Campus Teatinos, 29071, Málaga, Spain

{cipriano,anacm,jlblanco,jafma,jgonzalez}@ctima.uma.es

http://www.isa.uma.es

Abstract. Assistant robots like robotic wheelchairs can perform an effective and valuable work in our daily lives. However, they eventually may need external help from humans in the robot environment (particularly the driver in the case of a wheelchair) in order to accomplish safely and efficiently some tricky tasks for the current technology, i.e., open a locked door, traversing a crowded area, etc. This paper proposes a control architecture for assistant robots designed under a multi-agent perspective that facilitates the participation of humans into the robotic system and improves the overall performance of the robot as well as its dependability. Within our design, agents have their own intentions and beliefs, have different abilities (that include algorithmic behaviors and human skills), and also learn autonomously the most convenient method to carry out their actions, through reinforcement learning. The proposed architecture is illustrated with a real assistant robot: a robotic wheelchair that provides mobility to impaired or elderly people.

This work was supported by the Spanish Government under research contract CICYT05-01391.

1 Introduction

The assistant robotics field covers those applications where a mobile robot helps humans to perform certain tasks. Examples of robotic assistant applications are house cleaner and keeper robots, tour guiders, assistant robots for elderly people, etc., in which a robot must work within human environments interacting intelligently with people.

The first consideration to be taken into account in assistant robots is that they are designed to serve non-expert people who usually prefer to communicate and to interact with machines in the same manner they do with other people. Moreover, the presence of a robot within human scenarios like houses, offices, public facilities, etc., imposes a high degree of operation robustness and physical safety for humans as well as a sophisticated set of robot capabilities like maneuvering within narrow and/or crowded spaces, avoiding mobile obstacles, docking, etc. Planning tasks in such complex and typically large environments also represents a tough problem due to the large number of elements involved.

These issues are usually beyond the capabilities that the current technology offers, so the use of assistant robots operating autonomously within human environments is not yet extended. This lack of robot capabilities could be approached by considering the assisted person (or any other from the surroundings) as an additional component of an augmented robot. That is, humans can be integrated into the robotic system allowing it to extend its abilities through skills either not supported by the robot (i.e. take an elevator) or supported by the robot but in a different and (maybe) more secure manner (i.e. maneuvering in a complex situation). These skills may range from complicated low-level motions to high-level decision-makings. Obviously, the robot control architecture must be specifically designed to take into account this degree of interaction.

In the literature, mobile robotic architectures have considered the particular requirements of assistant applications from different perspectives. Some works ([1],[7],[11]) ensure desirable properties as robustness and fault tolerance in the architecture components by providing automatic software design tools, mechanisms to deduce safe actions before they are executed, techniques to check resource availability, etc. In the teleoperation area, whose applications can be seen close to assistant applications, collaborative control [12] is used to develop robotic architectures that support a tight relation between humans (expert operators) and machines. Through collaborative control, robots accept advises from operators to decide its actions. Such relation between human and robot improves the robot operating capacity but it restricts the human to physically act when the robot is not capable to continue its plan, for example when it must pass through a locked door. Other works also consider human-machine integration or cooperation ([12], [24], [30]) in a way that it enables a human to provide robot capabilities enough to operate in human environments. In these cases, human only serves as a command input provider.

Elsewhere a hybrid robotic architecture, called ACHRIN, specifically designed for assistant robots has been presented ([16],[17]). ACHRIN facilitates a non-expert person to be involved in the robotics operation when needed, through a special human-robot integration. Such an integration allows the human to participate at all levels, from deliberating a plan, to executing it, or even acting as an extra sensor. But in spite of those remarkable features, ACHRIN exhibits some shortcomings:

-Since ACHRIN is a tiered architecture, its components are communicated in a client/server fashion, and thus, communications should be built when designing the architecture. This characteristic prevents the system to be easily scaled up or dynamically modified, i.e., adding components.

-Considering a number of components that perform the same task (redundancy) imposes the necessity of selecting the best one according, for instance, to the past experience. In ACHRIN this selection is carried out by a component that implements a simple and hand-coded policy based on the previous results of execution. However, it would be desirable a more effective and flexible policy that could provide components with a certain degree of autonomy.

In this paper we propose a multi-agent re-design of ACHRIN to overcome the commented limitations. Multi-agent systems (MAS) is a subfield of AI that studies those complex systems formed by several agents that interact in some way. Although

MAS is a mature discipline, there is no general agreement on what an agent is, and several definitions can be found in the literature ([15],[28],[40]). However they all have a common aspect: autonomy, learning capabilities and rational behavior are essential aspects that agents must have. Following this, we have modified ACHRIN by turning all its components into autonomous agents. In our implementation, each agent possesses a mental state, that is, a set of intentions and beliefs, a group of abilities to perform a particular action, i.e., navigation, and a learning mechanism, Q-learning [22] in our case, to decide at each moment the best ability to accomplish the agent intentions.

MAS becomes a robust and scalable option when managing different elements with different goals that own some kind of information. Moreover, the use of a multi-agent framework seems more appropriate for an assistant robot application than deliberative, reactive or hybrid robotic architectures, since it permits a more natural integration of the human into the robotic system. We refer the reader to [40] for a complete discussion of multi-agent systems.

The rest of the paper is organized as follows. Section II gives an overview of the proposed multi-agent architecture. Section III details inter-agent communications. Section IV goes into the learning capabilities of agents to select autonomously the most appropriate abilities for performing their actions, and section V describes the application of the multi-agent architecture to a real assistant robot: a robotic wheelchair. Finally, some conclusions and future work are outlined.

2 Multi-Agent Control Architecture

The main feature of the multi-agent architecture presented in this paper is the integration of human abilities into the agents that constitute the architecture, as commented further on. In this section, a top-bottom description of the proposed architecture is provided: first, we focus on its structure, that is, the type of agents that includes, highlighting the differences with a non-agent-based architecture. Then, we detail the internal structure of each agent, the so-called Common Agent Structure (CAS).

2.1.- Structure of the Multi-Agent Architecture

One of the main characteristics that makes a multi-agent architecture (as well as any other agent-based system) different from other approaches is the interconnection between their constituent elements. A non-agent-based architecture, as the one shown in figure 1-left, establishes a fixed and normally one-to-one or one-to-few-ones connection between their components. In contrast, in a multi-agent approach (figure 1-right), the agents that make up the robotic architecture are fully interconnected and these connections are completely dynamic.

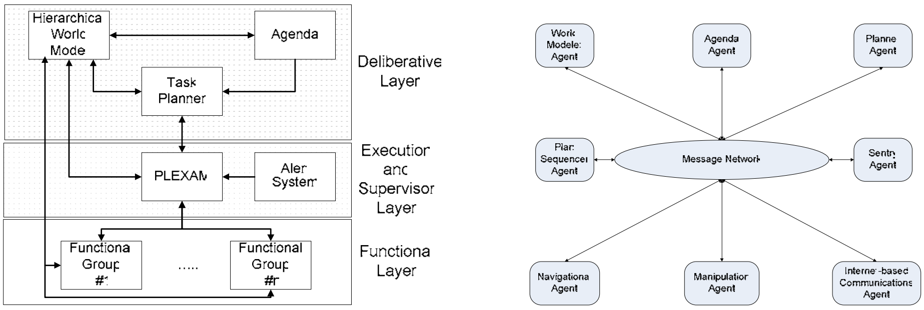


Figure 1. Left) A scheme of a hybrid robotic architecture (ACHRIN [16]). Right) Its evolution to a multi-agent system. Notice that in a client/server system (left) the addition of a component implies the modification of the components connected to it, which does not occur in a MAS. In our multi-agent re-design of ACHRIN we have considered navigation, manipulation and internet-based communication (access to email, for instance) as the functional tasks carried out by the combined human-robot system.

In particular, our multi-agent architecture, which is an evolution of ACHRIN (fig. 1-left), is called MARCA[1], and it is composed of the following agents:

-World Modeller Agent: This agent manages information stemmed from the environment in a human-like manner through the use of a mechanism called abstraction ([16],[17],[22]). It contains both geometric information, i.e. the position of a particular distinctive place, and symbolic information, i.e. a human-understandable label to refer to a room.

[1] MARCA stands for Multi-Agent-based Robotic Control Architecture.

-Agenda Agent: It accepts and manages requests from different users. Usually people will request the robot to execute some task, although it is also useful to accept requests from other robots, applications, etc. This enhances the robotic architecture with a higher and more intelligent interaction within its environment. The Agenda broadcasts to the rest of agents the necessity of achieving a goal.

-Planner Agent: This agent is capable of generating a plan to achieve a certain goal taking into account information from the world model as well as the available human+robot abilities. The resultant sequence of abilities that achieves the goal will be taken on by the correspondent agents. For instance, navigation actions will be carried out by the Navigational Agent. In our current implementation the Planner Agent runs a hierarchical, independent-domain algorithm for planning a variety of tasks [18].

-Sentry Agent. The Sentry Agent is responsible of checking unexpected dangerous situations in the system. It resembles human behavior since it reacts to external stimuli through both voluntary and reflex actions ([19],[16]). A reflex action is an automatic and involuntary movement, which is triggered by an external stimulus, i.e. the "shut the lids" reflex when an irritating substance enters the eye. On the other hand, a voluntary action is a conscious action, which is carried out in response to a stimulus, for example, we reduce the speed of our vehicle when approaching a red light. Notice that in human beings none of these actions are planned but instinctive. In fact, in neural science the border between reflex and voluntary actions is not well defined [26]. Voluntary and some reflex actions can be learnt and sometimes even inhibited, that is, a person can voluntarily ignore certain stimuli. In this sense, the Sentry Agent can ignore certain stimuli as human do, in order to adequate its behavior to the human preferences. For example, a collision stimulus can trigger a stop reflex action during navigation, but maybe the robot is carrying out a short-distance approach to some object. In such case, the stop reflex can be inhibited to achieve the proposed goal. The major concern when accepting an inhibition is the physical safety of humans and the robot. Thus, such an inhibition relies on an ad-hoc function that takes into account the human preferences, the current action that the robot is executing, and the robot state. Other architectures (non-agent-based) also include alert mechanisms ([3],[10]) but none of them provide the capability of consciously ignoring certain alerts from the environment.

-Plan Sequencer Agent. This agent sequences the actions that form a plan and manages the results of their execution. It is in charge of coordinating the different agents that have to perform the actions of the plan. It is also in charge of reacting properly to risky situations, for example stopping and blocking the vehicle when a collision or a critical battery level alert are reported.

The architecture is completed by a set of agents that physically perform actions. Each agent carries out a particular one, i.e. navigation or manipulation, by possibly using different abilities. For instance, the Navigational Agent can perform navigation through three abilities: a reactive algorithm, a path-planner algorithm, and a human manually guiding the vehicle. When an agent accounts for different abilities, it must decide at each moment which is the best one to accomplish its task. This decision process should be autonomous and independently carried out by agents, as we will show in section 4.

In our work we have considered three agents that respectively perform navigation, manipulation, and internet-based communication (i.e., email access). Some results are presented in section 5.

2.2.- Common Agent Structure

As it was stated before, the main feature of MARCA is that it distributes some of the abilities of the human among all the agents of the robotic architecture, in order to augment or to improve the functionality of the system. Thus, a human can help the robot to capture world information (as we have demonstrated previously in [10]), or to provide a plan for achieving a goal (interacting with the planner agent, as we have shown in [18]), or even to physically perform actions like open a door or call an elevator.

This human-robot integration is supported through the use of the *Common Agent Structure (CAS)* as a skeleton for designing every agent (see fig. 2). The CAS enables each agent to consider both human and robot capabilities to accomplish its work through the *skill units*, previously considered in ACHRIN. *Robotic skill units* represent abilities of the robot implemented by software algorithms, while *human skill units* represent the connection of the agent to the human, enabling people to perform actions through appropriate interfaces (i.e., voice communication). An example of the inclusion of human abilities is the case of the Navigational Agent devoted to perform robot navigation: it contains a set of robotic skill units to navigate between two locations, i.e., a reactive navigator and a path-tracker algorithm, but it can also consider the human participation through a human skill unit that permits him/her to manually guide the robot. Thus, the agent accounts for three different ways of accomplishing actions like "*go to place X*".

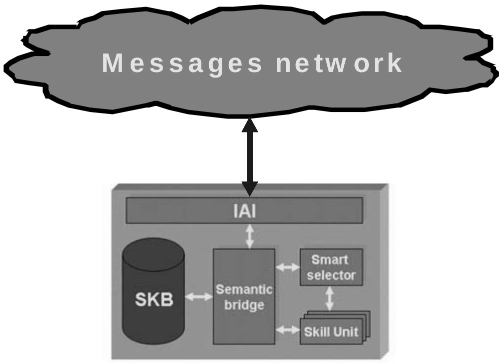


Figure 2. The common agent structure (CAS). All the agents in MARCA are designed using this structure. The number of skill units contained into each agent is variable and it depends on the agent's functionality as well as on the human and robot capabilities provided by the CAS.

One of the commonly accepted features that agents must possess –along with autonomy, proactiveness, or communications- is *learning*. So, among the different possibilities that an agent accounts for accomplishing its tasks, it should be able to learn over time the most convenient one, given its past experience and the current conditions of the environment [31]. Within the CAS skeleton, the *Smart Selector* (SS) is in charge of the agent's learning process, and it is always active in order to adapt the agent performance with respect to variable environmental conditions, as commented in more detail in section 4.

All skill units considered by the SS to accomplish a requested action pursuit the same goal (i.e. to reach to a destination in the case of navigating), but they may exhibit differences in the way they are invoked, especially in the case of human units. For instance, the navigation action "go to p1"[2] can be carried out by invoking a reactive navigation algorithm (a robotic skill unit) in the form *"navigate to (x=0.2, y=12.3, phi=12)"*, while the human skill unit would require a linguistic label that represents the destination in terms of the human knowledge, i.e. *"go to Peter's office"*.

The translation into the skill unit parameters from an action requested by any other agent in the architecture is carried out by the *Semantic Bridge* (SB). This component also permits the internals of the agent (Smart Selector and skill units) to request/provide data to other agents of the architecture (that is, it deals directly with the semantics explained in section 3). For instance, the previously commented human skill unit can request an external agent for the linguistic label of the destination *p1*.

The Semantic Bridge is also in charge of maintaining the *Semantic Knowledge Base* (SKB) that represents the internal mental state of the agent in terms of intentions and beliefs. Such an internal mental state is updated by the SB with incoming messages or requests from other agents. Communications between agents relies on the *Inter-Agent Interface* (IAI) that provides communication primitives to send/receive messages following a fixed semantic, as commented in more detail in section 3.

3 Inter-Agent Communication

In this section we focus on how agents communicate with each other. Since agents are autonomous entities with their own motivations, they cannot be forced by other agents to do some action; hence, communication is used to influence or persuade them to perform tasks.

In contrast with traditional software architectures, in which communication between different pieces of software usually makes intensive use of the client-server paradigm (just using procedural calls, as in our previous architecture ACHRIN, RPC, or any other function invocation mechanism), MAS imposes a much richer set of interactions between agents. Thus, if communications are expected to affect the internal mental states of agents, they should not be mere raw data, but information about agent attitudes.

In our approach, inter-agent communications are based on *message passing*, which is the most common scheme for MAS. In particular, we use the standardized message

[2] *p1* is a symbol stored in the internal world model of the robot that represents a particular

format proposed by FIPA, namely ACL [14]. We have been highly inspired by the CAL specification [13] in order to define the underlying semantics of the communicative acts.

Mental attitudes of agents in our architecture are defined using these operators:

- *Beliefs*: the operator $B_i\ p$ means that agent *i* beliefs that fact *p* holds.
- *Desires*: the operator $D_i\ a$ means that action *a* is a goal to be achieved by agent *i*.
- *Intentions*: the operator $I_i\ a$ means that agent *i* is intent on carry out the action *a*. The semantic meaning of intention differs from the concept of *goal* normally used in robotics; intentions are used in MAS for invoking the execution of actions.

Moreover, to model actions we need the following additional operators:

- *Agent*(*i*,*a*): Means that agent *i* is capable of performing the action *a*.
- *Done(a)*: Represents that action *a* has been carried out.
- *Feasible(a)*: This means that, for a given instant of time, preconditions for executing action *a* are met.

The set of communicative acts or *performatives* that we use in MARCA and their associate preconditions, triggers, and effects are detailed in Table 1. There we have used the prefixes +/– to designate the insertion/removal of facts from the agents SKB. Since our agent execution model is event driven, these modifications in an agents'

location of its workspace. Symbols are managed by planner agents to plan robot tasks.

mental state trigger actions into the IAI and the semantic bridge, for instance, the execution of another communicative act or the initiation of a given skill unit.

The meaning of the specified performatives is explained as follows:

- **Inform**: Implies the intention of an agent *i* of letting another agent *j* to know something that *i* currently believes. We assume a sincere behaviour of agents, that is, the content communicated by an agent represents its current attitudes which are immediately incorporated to the beliefs of the receiver agent.
- **Request**: Sender agent requests the receiver to execute a given action *a*. The target agent has to decide whether accept or refuse the proposal. As shown in table 1, the SKB of the target agent is consequently updated to reflect the new intention of performing the action (in case of acceptance), or the desire of communicating the rejection to the sender agent in other case.

Performatives	**Sender (Agent *i*)**			**Receiver (Agent *j*)**
(sent from *i* to *j*)	**Trigger**	**Preconditions**	**Effects in SKB**	**Effects in SKB**
inform(j,ϕ)	$+D_iB_j\,\phi$	$B_i\,\phi$	$-D_iB_j\phi$	$+B_j\,\phi$
request(j,a)	$+D_i$ Done(a)	$\neg B_i$ Agent(i,a) $\wedge$ B_i Agent(j,a) $\wedge$ Preconditions(a) $\wedge$ $\neg B_iB_jD_i$ Done(a)	$+B_iB_jD_i$ Done(a)	$+B_jD_i$ Done(a) *Accept:* $+I_j$ Done(a) *Reject:* $+D_jB_i\neg I_j$ Done(a)
agree(j,a)	$+I_i$ Done(a)	B_iD_j Done(a) $\wedge$ $\neg B_iB_jI_i$ Done(a)	$+B_iB_jI_i$ Done(a)	$+B_jI_i$ Done(a) $+B_j$ Feasible(a)
refuse(j,a)	$+D_iB_j\neg I_i$ Done(a)	B_iD_j Done(a) $\wedge$ $\neg B_i B_j\neg I_i$ Done(a)	$+B_iB_j\neg I_i$ Done(a)	$+B_j\neg I_i$ Done(a)
cancel(j,a)	$-D_i$ Done(a)	$B_iB_jI_i$ Done(a) $\wedge$ $\neg$Done(a)	$-B_iB_jI_i$Done(a)	$-B_jD_i$ Done(a) *And probably*: $-I_j$ Done(a)
failure(j,a)	$-I_i$ Done(a)	$B_iB_jI_i$ Done(a) $\wedge$ $\neg$Done(a)	$-B_iB_jI_i$ Done(a)	$+B_j\neg I_i$ Done(a) $+B_j\neg$Feasible(a)
query(j, β)	$+D_i$ Done(Eval(β))	$\neg B_i$Agent(i,Eval(β))$\wedge$ B_iAgent(j, Eval(β))	―――――	$+D_jB_i$ Eval(β)

Table 1: The communication acts that we use in our system to define the semantics of inter-agent communications. The nomenclature is explained in detail in the text.

- **Agree**: The agreement of an agent *i* to perform a requested action for some other agent *j*. This communicative act is triggered on acceptation for a request.
- **Refuse**: Agent *i* refuses the request of agent *j* for performing the action *a*.
- **Cancel**: Informs to an agent *j* that agent *i* has no longer the intention of agent *j* of performing the action *a*. As a consequence, the receiver agent will leave its

intention of carrying out the action, if there is no other agent that still desires its execution.

- **Failure**: Agent *i* aims to perform action *a* for another agent *j*, but it was not possible to complete the execution and currently the agent does not longer intent on trying it.
- **Query**: This performative contains an expression β, whose meaning is not standardized, but it is assumed that the receiver agent should be able to evaluate it. In the preconditions of the performative, queries are sent only when the agent cannot solve the expression, thus communications are used only when the agent need external information.

In order to pass messages between agents, the performatives are sent as formatted strings, which are coded as an ACL compliant plain text message. In section 5 we will show examples of such messages.

4 Learning Capabilities of Agents

One of the main characteristics that defines an agent is its capability for learning from its own experience. In MARCA, agents must learn how to select the appropriate skill unit (either human or robotic) when carrying out their actions. For that purpose we have chosen Reinforcement Learning (RL) to be implemented in our CAS Smart Selector, since RL has a thoroughly studied formal foundation and it also seems to obtain good results when applied to mobile robotics tasks ([27],[35]).

RL has been analysed widely elsewhere; the classical survey by Kaelbling, Littman and Moore [22] is a good starting point for a deeper research into the subject [36]. In short, RL is a machine learning paradigm where an agent in a state *s* executes some action *a* turning its state into *s'* and getting a reinforcement signal or reward *r*. Those experience tuples (*s*,*a*,*s'*,*r*) are used for finding a policy π that maximizes some long-run measure of reward. It is supposed that the environment where the agent evolves is a non-deterministic one, which means that taking the same action in the same state on different occasions may yield different next states and/or rewards. RL is graphically explained in figure 3:

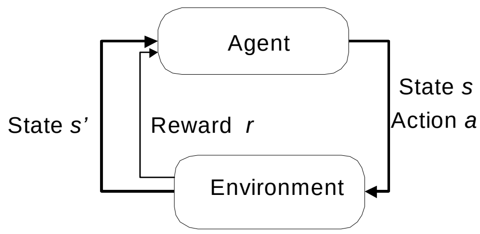


Figure 3. The Reinforcement Learning framework implemented in our CAS Smart Selector. Agents learn their skill unit selection policy from their interaction with the environment. The execution of a certain action (skill unit) produces a reward value that guides the learning process over time.

There is a comprehensive bunch of methodologies for modeling and solving RL problems. A well-known solution are *Markov Decision Processes* (*MDPs*), that can be defined by:

- a set of states *S*,
- a set of actions *A*,

- a reward or reinforcement function $R : S \times A \rightarrow \Re$ meaning that if the agent is in state *s* and performs action *a*, it receives a *r* reinforcement signal.
- a state transition function $T : S \times A \rightarrow$ *Π(s)*, where *Π(s)* is a probability distribution over the set *S*. So, probability of making a transition from current state *s* to next state *s'* executing action *a* is written as *T(s,a,s')*.

Upon these sets and functions, an optimal policy π that maximizes the obtained reward can be computed in this way:

- The *optimal value* of a state $V^*(s)$ is the expected reward that the agent will gain if it starts in that state and executes the optimal policy. This optimal value is stated as the sum of the expected reward R plus the expected discounted value of the next state, using the best available action:

$$V^*(s) = \max_a (R(s,a) + \gamma \sum_{s' \in S} T(s,a,s')V^*(s')), \forall s \in S \ . \qquad (1)$$

where parameter γ is a *discount factor* that represents how much attention is paid to future rewards.

- The optimal policy then is specified by expression:

$$\pi^*(s) = \arg\max_a (R(s,a) + \gamma \sum_{s' \in S} T(s,a,s')V^*(s')) \ . \qquad (2)$$

Both reinforcement and state transition functions are named a *model*. However, such a model is not always known in advance; in fact, most of robotics applications cannot provide that prior knowledge. In these situations, there is an approach that can be used for learning the optimal policy: *Q-learning*. Q-learning uses the following optimal value function Q^*:

$$Q^*(s,a) = R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') \max_{a'} Q^*(s',a') \qquad \textbf{(3)}$$

Since the Q-function makes the action explicit, it can be recursively computed on line by means of:

$$Q(s,a) = Q(s,a) + \alpha(r + \gamma \max_{a'} Q(s',a') - Q(s,a)) \qquad \textbf{(4)}$$

Parameter α is the *learning rate*, and it must slowly decrease in order to guarantee convergence of function Q [5]. Once the optimal Q-function Q^* is obtained, the optimal policy can be fixed as it was stated in (2).

For illustrating how Q-learning performs the learning proccess we focus our attention on the Navigational Agent. This agent is in charge of the mobility issues of the robot: it must provide a safe displacement of the vehicle in its environment as it reaches some given spatial references. As it was previously commented, there are three skill units available in the NA: a human skill unit that puts human in charge of movement through a joystick, a path tracker algorithm that follows a previously calculated path, and a robotic reactive skill unit that implements a reactive navigation algorithm [2]. The agent learns via Q-learning an optimal policy that states which skill unit is the best navigation option at every moment.

The mathematical formulation of the Q-learning technique for the NA is as follows:

- Every state is built as the sum of five factors, namely: unexpected obstacles detection (O), local minimum detection (LM), critical obstacles detection - like doors, ramps, etc. that demand human intervention- (CO), available robot energy (RE), and human energy (HE). Boths obstacle detection (O and CO) can be implemented by means of laser range sensors. The presence of a local minimum (LM) is detected when the robot is not able to leave a section of the route for a certain time. This is a typical drawback of reactive navigation [39]. Finally, energy levels are measured in terms of battery levels (RE) and human fatigue (HE) respectively; human fatigue is due to the effort done by the human since he/she has to be alert during the navigation in order to avoid any hazardous unexpected situation. This fatigue can be measured through the frequency of actuations on the joystick and the frequency of alerts produced by the human within the Sentry Agent. The first three factors (O, LM, and CO) have been discretized into yes/no values, whereas energy factors (RE and HE) have been discretized into low/medium/high values. The result of this discretization is a set of 72 possible states in the evolution of the navigational system.

- The set of actions *A* simply matches the set of skill units of the agent, since the policy to learn looks for the best sequence of skill units to apply. In this way, the possible actions are HUMAN, DELIBERATIVE, and REACTIVE navigation.

- The reinforcement function *R* favours states with a high level of available energy along with actions that can be applied with a low human-robot interaction effort. In this way, a HUMAN action requires a high interaction effort since it needs to activate the voice communication with the human; the REACTIVE action does not need any interaction; and the DELIBERATIVE action would be an in-between that just demands some kind of path/speed planification. The final goal of this definition of reward is to obtain policies with an energetic and interaction efforts as low as possible.

In the following section, the results of some experiments conducted on the Navigational Agent for evaluating our RL approach are shown.

5 A real robotic assistant application

MARCA has been tested on an assistant robot called SENA (see fig. 4). It is a robotic wheelchair based on a commercial powered wheelchair that has been equipped with several sensors and an onboard computer to reliably perform high-level tasks in indoor environments. SENA accounts for Wi-Fi connection capabilities that improve to a great extend the possibilities of our robot, ranging from tele-operation of the vehicle to the human driver access to internet.

Figure 4. Two views of the SENA robotic wheelchair in which we have evaluated our human-robot integration architecture.

Our tests have been carried out within our lab and near corridors, in which the user can select a destination via voice. Since the main operation of SENA is navigation, we have focussed our experiences on the Navigational Agent; an example of a navigation task around our lab environment can be seen in figure 5.

In a navigation task, the Planner Agent interacts with the user (as described in [18]) for constructing the best route to the goal, and then sends it to the NA via the Messages Network using the semantics described in section 3. When a navigation step arrives at the agent, its Semantic Bridge translates it to the Smart Selector in order to perform the action. Then, the Smart Selector applies the Q-learning algorithm, and the best skill unit learnt until that moment will be activated.

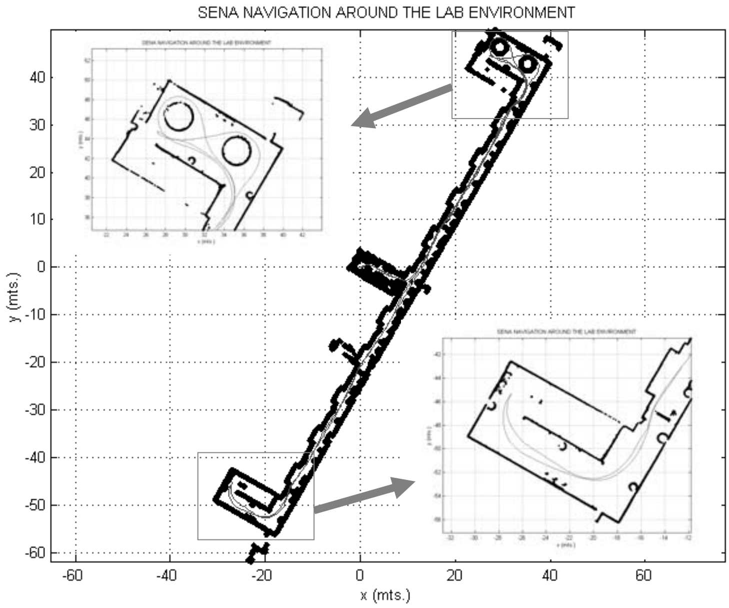


Figure 5. An example of SENA navigation in the surroundings of our laboratory. Rooms and corridors are depicted in thick lines, whereas the path followed by SENA is shown in thin lines. For illustrating the variety and difficulty of the paths performed by SENA in this environment, two interesting zones have been zoomed.

Our results reveal that Q-learning is effectively working and provides an important degree of autonomy in the selection of each agent's abilities, producing good policies for navigational issues. We have compared the reward obtained by Q-learning against the reward obtained by an intuitive algorithm that selects a supposed best skill unit, implemented through a common-sense approach. Basically, the intuitive algorithm selects HUMAN action whenever a critical obstacle comes up; it selects DELIBERATIVE action if neither unexpected obstacles nor local minima are found; and finally, uses REACTIVE action in every other situation. The comparison is plotted in figure 6.

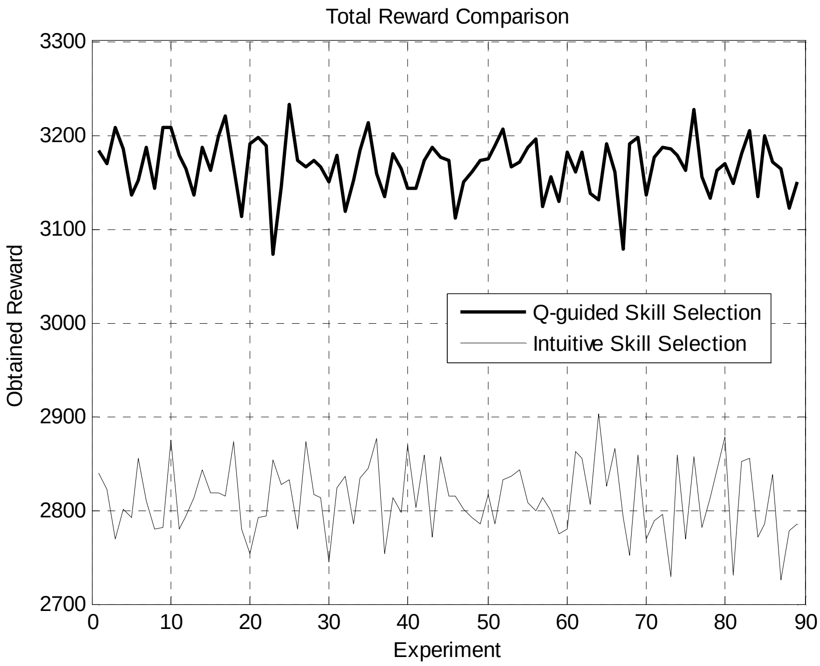


Figure 6. Q-learning (solid line) vs. Intuitive algorithm (dashed line) rewards. The policy found by Q-learning offers better results than using an intutive algorithm.

As it can be seen, Q-learning finds a policy of actions that gets better rewards than the intuitive algorithm: average rewards are 3167 (Q) and 2812 (int), and standard deviations are 29.4 and 37.9 respectively, which makes Q-learning an outstandingly better option.

Furthermore, Q-function really converges to a certain value after a proper number of learning steps. Figure 7 shows, for two states (expressed as a combination of the five factors mentioned in section 4), how Q-function of every action (HUMAN, DELIBERATIVE and REACTIVE) finally converges. It can be seen that, though both states are similar in terms of unexpected obstacles, local minima and critical obstacles, their different energy levels lead to a different skill unit selection.

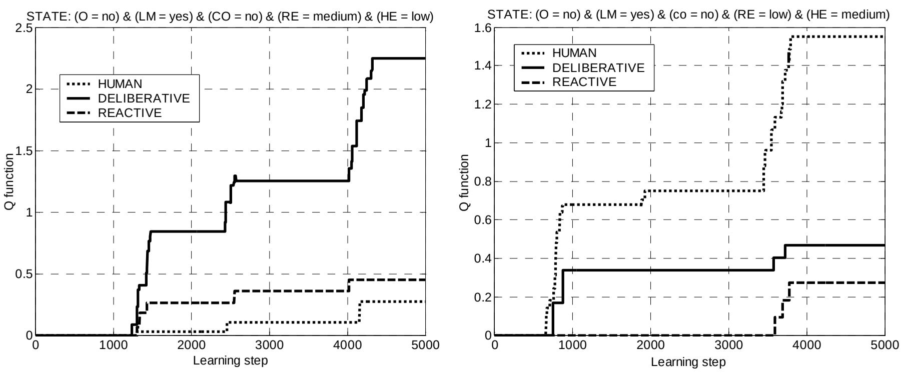
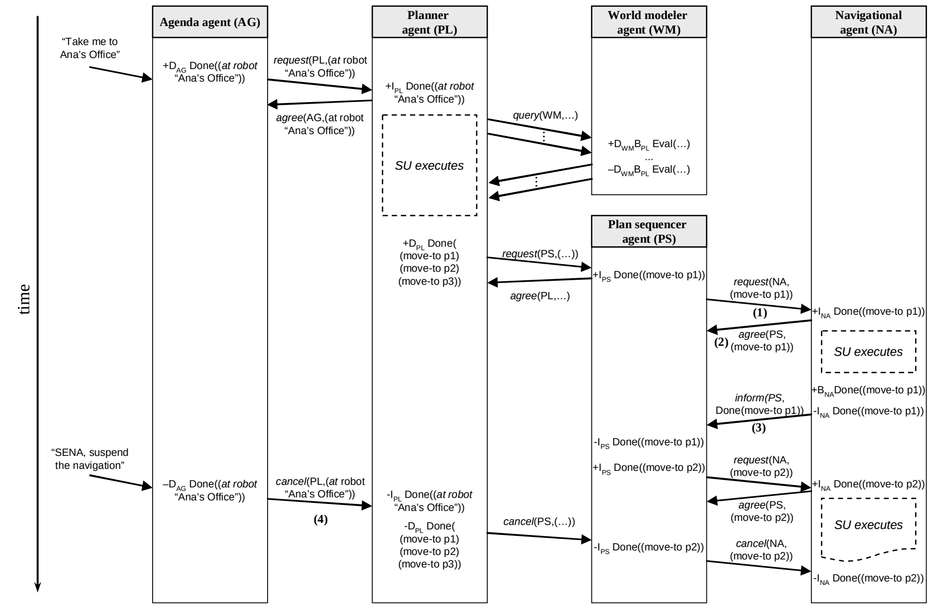


Figure 7. Q function values for two selected states. The selected best actions are DELIBERATIVE and HUMAN actions, respectively.

Regarding the internal state of agents and how communicative acts are used between them, please refer to figure 8, where the main changes in agents's mental state are shown to illustrate the evolution of their attitudes through an experiment with SENA. Some messages corresponding to key communicative acts are also sketched as arrows between agents. We can interpret the semantic meaning of the agents attitudes as follows. At first, the user enters a navigation command to the robot via voice, which is captured by a SU inside the Agenda agent as a new desire: D_{AG} *Done((at robot "Ana's Office"*)). This complex task must be decomposed by the Planner agent, thus the user command is requested to this agent, which agrees and introduces into its mental state the intention of carrying out this action. This event triggers the execution of an internal SU which queries the World Modeller agent the abstract information needed to plan the complex action. Once that a plan consisting of elementary actions is available, a new desire in the planner agent's SKB for executing the plan leads to a request to the Plan Sequencer agent, which also agrees and incorporates the plan to its intentions. This agent therefore sends subsequent requests to the Navigational Agent, whose SS will select the most promising SU for performing the demanded navigation at each instant of time through the previous Q-leaning approach. In this experiment the user asks the robot to abort the navigation while the second elementary action is being executed. As can be seen in figure 8, the user annulment of the navigation task produces a chain of *cancel* messages between agents, as well as the retreat of intentions from their SKB. As a result, all the involved agents return to their previous states and the navigation action becomes definitively forgotten at all the abstraction levels, from high-level path-planning to low-level local navigation algorithms. Finally, we should highlight that messages between agents has been implemented as ACL plain-text messages, as can be seen with the messages denoted as (1)-(4) in figure 8. We use the standard fields and an additional timestamp field, which can be employed to measure communication delays, among other possible uses.



(1)
```
(request
   :sender (agent-identifier PlanSequencerAgent)
   :receiver (agent-identifier NavigationAgent)
   :content "(move-to ((x -0.45)(y 10.34)))"
   :reply-with req1
   :X-Timestamp 2006/1/20,18:22:02.699
)
```

(2)
```
(agree
   :sender (agent-identifier NavigationAgent)
   :receiver (agent-identifier PlanSequencerAgent)
   :content "(move-to ((x -0.45)(y 10.34)))"
   :in-reply-to req1
   :X-Timestamp 2006/1/20,18:22:03.234
)
```

(3)
```
(inform
   :sender (agent-identifier NavigationAgent)
   :receiver (agent-identifier PlanSequencerAgent)
   :content "Done(move-to ((x -0.45)(y 10.34)))"
   :X-Timestamp 2006/1/20,18:23:25.509
)
```

(4)
```
(cancel
   :sender (agent-identifier AgendaAgent)
   :receiver (agent-identifier PlannerAgent)
   :content "(at robot \"Ana's Office\")"
   :X-Timestamp 2006/1/20,18:23:58.354
)
```

Figure 8. The schematic representation of agents' internal states and some messages sent during the experiment described in the text.

5 Conclusions and future work

Robots that operate in human environments, especially assistant robotic applications, may allow the researches to relax somehow the typical autonomy requirements of a conventional robotic platform. However, this leads to a greater amount of work on human-robot interaction issues. Commonly, human-robot interaction is considered just as a simple communication between the robot architecture and the human. However, we believe that assistant robotics needs a much stronger interaction if the robot has to face successfully complex situations, especially those that it cannot solve by its own means. Thus, the human must be integrated into the robotic architecture as a part of an augmented system, in order to achieve the so-called *human-robot integration* that we claim in our work.

We have presented in this paper MARCA, a multi-agent robotic architecture that enables such human-robot integration. We have chosen a multi-agent system approach rather than other conventional architectures because agents are closer to represent the human as a part of the system than other software constructions (modules, procedures, objects, etc.). Agents, as well as humans, have intentions, mental states, and use some semantics in their communications. In addition, they have some learning capabilities that allow them to adapt to environmental changes and to achieve good performances over time. Furthermore, MAS systems also offer another valuable benefits, like robustness or scalability.

We have defined the semantics for the agents to communicate and maintain their internal mental states, as well as how this semantics is translated into practical requests for the algorithms of the architecture. Besides, since any agent in MARCA is endowed with a set of robotic and/or human skill units for performing some action, it must decide which of them is the best in every situation. We have implemented a Q-learning procedure that learns this association over time, trying to optimize the long-term behavior of the system.

MARCA has been applied to a real robotic wheelchair, and the suitability and effectiveness of the architecture to this kind of applications have been experimentally validated.

This is not a finished work, and these preliminary results form the basis of an on-going work. In fact, this paper is a first step towards the inclusion of the human as an agent within the architecture, since at this stage the human is not yet an agent, but his/her capabilities are distributed inside any agent that needs them for augmenting its skills. Furthermore, we continue with our work with assistant robots (such as our robotic wheelchair SENA, or our recent tour guider robot SANCHO) in order to implement solutions for important requirements of assistant applications.

References

1. Alami R., Chatila R., Fleury S., Ghallab M., Ingrand F. An Architecture for Autonomy. Int. J. Of Robotics Research. Special issue on Integrated Architectures for Robot Control and Programming. 1998
2. Blanco J.L, González J., Fernández-Madrigal J.A. The PT-Space: A new Space Representation for Non-Holonomic Mobile Robot reactive Navigation. Tech. Report, University of Málaga, 2005.
3. Brugali D. and Fayad M. E. Distributed Computing in Robotics and Automation. IEEE Transactions on Robotics and Automation, vol. 18, no. 4, 2002.
4. Cohen P.R. and Levesque H.J. Intention is choice with commitment. Art. Intellig., vol. 42, n. 3, 1990.
5. Even-Dar E. and Mansour Y. Learning Rates for Q-learning. Journal of Machine Learning Research 5 , 2003, pp. 1-25.
6. Fernández J.L. and Simmons R. G. Robust Execution Monitoring for Navigation Plans. Procs. of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Victoria, B. C., Canada. 1998.
7. Fernández-Madrigal J. A. and González J. A Visual Tool for Robot Programming. 15th IFAC World Congress on Automatic Control, Barcelona, Spain, 2002.
8. Fernández-Madrigal J. A. and González J. Multihierarchichal Graph Search. IEEE Trans. on PAMI. Vol. 24. No. 1, January, 2002.
9. Fernández-Madrigal J. A. and González J. Multi-Hierarchical Representation of Large-Scale Space. International Series on Microprocessor-based and Intelligent Systems Engineering. Vol. 24, Kluwer Academic Publishers, Netherlands, 2001.

10. Fernández-Madrigal J. A., Galindo C., González J. Assistive Navigation of a Robotic Wheelchair using a Multihierarchical Model of the Environment. Integrated Computer-Aided Engineering, vol 11, pp. 309-322, 2004.
11. Fleury S., Herrb M., Chatila R. GenoM: A Tool for the Specification and the Implementation of Operating Modules in Distributed Robot Architecture. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'97), Grenoble, France.
12. Fong T.W. and Thorpe C. Robot as Partner: Vehicle Teleoperation with Collaborative Control. Procs. of the 2002 NRL Workshop on Multi-Robot Systems, 2002.
13. Foundation for Intelligent Physical Agents. ACL Communicative Act Library Specification, 2002.
14. Foundation for Intelligent Physical Agents. ACL Message Structure Spec., 2002.
15. Franklin S. and Graesser A. Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents. Procs. of the 3rd International Workshop on Agent Theories, Architectures and Languages, Springer-Verlag, 1996.
16. Galindo C., González J., and Fernández-Madrigal J.A. A Control Architecture for Human-Robot Integration. Application to a Robotic Wheelchair. IEEE Systems, Man and Cybernetics Part B. (To appear)
17. Galindo C., González J., Fernández-Madrigal J. A. An Architecture for Close Human-Robot Interaction. Application to Rehabilitation Robotics. IEEE Int. Conf. on Mechatronics and Automation (ICMA'2005), Ontario (Canada), July-Agust 2005.
18. Galindo C., Fernández-Madrigal J. A., González J. Hierarchical Task Planning through World Abstraction. IEEE Trans. on Robotics, 20(4), pp. 667-690, 2004.

19. Galindo C., González J. And Fernández J. A. Interactive Task planning through múltiple abstraction: application to assistant robotics. 16th European Conf. on Artificial Intelligence, Valencia, Spain, August 2004.
20. Harnand S. Psychological and Cognitive Aspects of Categorical Perception: A Critical Overview. In Categorical Perception: the Groundwork of Cognition, Harnan S. (ed.), New York, Cambridge University Press, Chapter 1, 1987.
21. Hirtle S. C., and Jonides J. Evidence of Hierarchies in Cognitive Maps. In Memory and Cognition, Vol. 13, No. 3, 1985.
22. Kaelbling L.P., Littman M.L., Moore A.W. Reinforcement Learning: A Survey. Journal of Artificial Intelligence Research 4 (1996) 237–277.
23. Kandel E. And Schwartz J. H. Principles of Neural Science. Chapter 33. Elsevier Science Publishing Co. Inc., 1985.
24. Karim T. A Semi-Autonomous Reactive Control Architecture. Journal of Intelligent and Robotic Systems, nº 32, 2001.
25. Khatib O. Human-Centered Robotics and Haptic Interaction: From Assistance to Surgery, the Emerging Applications. 3rd Int. Workshop on Robot Motion and Control, 2002.
26. Kuipers B. J. The Cognitive Map: Could it Have Been Any Other Way?. Spatial Orientation: Theory, Research and Applications. Picks H. L. And Acredolo L. P., Plenum Press, New York, 1983.
27. Liu J. and Wu J. Multi-Agent Robotic Systems. CRC Press, 2001.
28. Maes P. Modeling adaptative autonomous agents. In C. G. Langton, editor, Artificial Life: An Overview, pp. 135-162, The MIT Press, Cambridge, MA, 1995.
29. Morioka K., Lee J.H., Hashimoto H. Human Centered Robotics in Intelligent Space. Proc. of the IEEE ICRA'02, Washington, DC.

30. Morris A.C., Smart C.K., Thayer S.M. Adaptative Multi-Robot Multi-Operator Work Systems. Procs. of the 2002 NRL Workshop on Multi-Robot Systems, 2002.

31. Murch R. and Johnson R. Intelligent software agents. Prentice Hall 1999.

32. Prochazka A., Clarac F., Loeb G. E., Rothwell J. C. and Wolpaw J. R. What do reflex and voluntary mean? Modern views on an ancient debate. Springer-Verlag. Nov. 1999.

33. Remolina E., Fernández J. A., Kuipers B. J., González J. Formalizing Regions in the Spatial Semantic Hierarchy: An AH-Graphs Implementation Approach, in Lecture Notes in Computer Science, Vol. 1661, pp. 109-124, 1999.

34. Scholtz J. Theory and evaluation of human-robot interaction. Proc. of the 36th International Conference on System Sciences, Hawai, 2003.

35. Smart W.D. and Kaelbling L.P. Effective Reinforcement Learning for Mobile Robots. International Conference on Robotics and Automation (2002)

36. Sutton R.S. and Barto A.G. Reinforcement Learning: An Introduction. The MIT Press, Cambridge, MA (1998).

37. Stone P. and Veloso M. A Survey of Multiagent and Multirobot Systems. In Robot Teams: From Diversity to Polymorphism, pp. 37-92. Edited by T. Balch and L. E. Parker. 2002.

38. Tahboub K.A. A Semi-Autonomous Reactive Control Architecture. J. Intell. Robotics Syst., vol. 32, n. 4, pp. 445-459, 2001.

39. Ulrich, I. and Borenstein, J. VFH+: Reliable obstacle avoidance for fast mobile Robots. Proc. IEEE International Conference on Robotics and Automation, 1998.

40. Wooldridge M. An Introduction to MultiAgent Systems. John Wiley & Sons Ltd. 2002.