Multi-Hierarchical Interactive Task Planning. Application to Mobile Robotics

Cipriano Galindo, Juan-Antonio Fernández-Madrigal, and Javier González, *Member, IEEE*

***Abstract*—Up to date, no solution has been proposed to human-machine interactive task planning that deals simultaneously with two important issues: i) the capability of processing large amounts of information in planning (as it is needed in any real application), and ii) being efficient in human-machine communication (a proper set of symbols for human-machine interaction may not be suitable for efficient automatic planning, and *vice versa*). Here we formalize a symbolic model of the environment for solving in a natural form these issues through a human-inspired mechanism that structures knowledge in multiple hierarchies. Planning with a hierarchical model may be efficient even in cases where the lack of hierarchical information would make it intractable. But in addition, our multi-hierarchical model is able to use the symbols that are most familiar to each human user for interaction, thus achieving efficiency in human-machine communication without compromising task-planning performance. We formalize here a general interactive task planning process which is then particularized to be applied to a mobile robotic application. The suitability of our approach has been demonstrated with examples and experiments.**

***Index Terms*—Interactive Task Planning, World Modeling, Hierarchical Task Planning, Mobile Robots.**

I. Introduction

There are a number of works in literature that provide automatic planning processes with the capability of interacting with humans ([49],[50],[51],[52]). They are mostly based on sharing with the human a set of symbols that represent the environment, usually the same set that is used for the task planning process. Sharing such a model with the human facilitates the subsequent sharing of intentions (as goals to achieve), processes (planning), and operating results (for guiding, acceptance, or correction of the plans). However, the reported symbolic representations, and specially the symbols needed for these applications may not be the most adequate for being understandable by every person or the best for planning efficiently (the real world can be categorized into different sets of symbols, some allowing for planning more efficiently than others, and some with no semantics for a human).

Another important issue in the use of a symbolic representation for operating in the environment is the *symbol grounding problem* ([2],[4],[3]), which involves the maintenance of dynamic associations between symbols and their real world counterparts. Out of the scope of interactive task planning, some works have addressed this, being a relevant approach the *anchoring* framework proposed in [1], that copes with the symbol grounding problem by maintaining links between symbols and sensor data that refer to physical objects.

Finally, there is a third problem related to cognitive human-machine interaction that is rarely taken into account when a symbolic representation of the environment is used: most of the algorithms that process the information, specially task planning, become very inefficient and even intractable when dealing with a realistic amount of data. We have found that this is an important problem in real environments [6] even when they are not large-scale, for example, when they contain a non-negligible number of objects that must be taken into account by the deliberative machine. This is particularly important in robotic applications.

In this paper we focus on sharing cognitive information, in particular task planning and categorization of the physical world, between the human and a planning system in general. Although planning systems may belong to a variety of application fields or be executed on very different platforms, e.g. a PC, a workstation, or an embedded system, in this paper we focus on the robotics arena, and consider the planning system of a mobile robot. Thus, our approach should be seen as a particular case of Human-Machine Interaction instead of Human-Robot Interaction since we will relax the interfacing requirements, assuming that the components that permit the machine to communicate to humans (verbally, mechanically, visually, etc.) or to perform physically are already given.

Manuscript received October 11, 2006. This work was supported by the Spanish Government under research project CICYT- DPI2005-01391. All the authors are with the System and Engineering Dpt. Of the University of Málaga (Spain) (corresponding author's phone: +34 952132892; fax: +34 952133361; e-mail: cipriano@ctima.uma.es).

In the last years research on robotics has found several areas of interest apart from the classical search for predictable, autonomous operation. Maybe one of the most promising is to consider the robot as a semi-autonomous agent that is to be embedded into a human social environment, which involves considering human-robot interaction as a non-negligible aspect. A number of important issues have to be solved for constructing such robots: reliability, long-term operation, real-time performance, sharing of information and skills with humans, etc. There are a variety of them dealing explicitly with human-robot interaction, since it can be studied ranging from the hardware-software interfaces that permit the robot and the human to physically communicate to each other ([35],[36],[37],[38]) to computational mechanisms for sharing cognitive information and processes ([29],[40]), emotions ([30],[39]), also including intermediate levels such as assistance in basic skills, like navigation in the case of assistant wheelchairs ([8], [31],[34]), manipulation ([28]), object delivering ([43],[44]), and many others ([45],[46],[47],[47]). Recently, some robotic architectures have been proposed ([5],[32],[33]) to provide the researchers with a suitable basis to address this diversity.

For addressing the interactive task planning issues commented above, we propose an approach based on the inclusion into a symbolic graph model of the concept of *abstraction* ([9],[27],[19]), that is, the structuring of symbols into hierarchies of detail, in such a way that task planning can use the different levels of abstraction as an heuristic to reduce computational cost and even make some intractable problems tractable ([6],[10]). We have found that abstraction is also specially relevant and suitable for addressing human-robot interaction at a cognitive level, and there has been psychological evidence that humans also use it, for example to structure our cognitive maps[1] ([4],[11],[12],[13]).

However, we also have found [9] that the use of a single hierarchy of abstraction for reducing the complexity of a given operation has an aspect that is often neglected: a given hierarchy may be suitable to solve a specific task (for example task planning) but not provide an effective solution to others (in our case, interacting with humans through understandable symbols). Thus we explore here the use of more than one hierarchy in the same model, what we call *multiple abstraction*, which has previously demonstrated important improvements in tasks like graph search ([9]). There is also psychological evidence on the use of multiple abstraction by human beings ([17]).

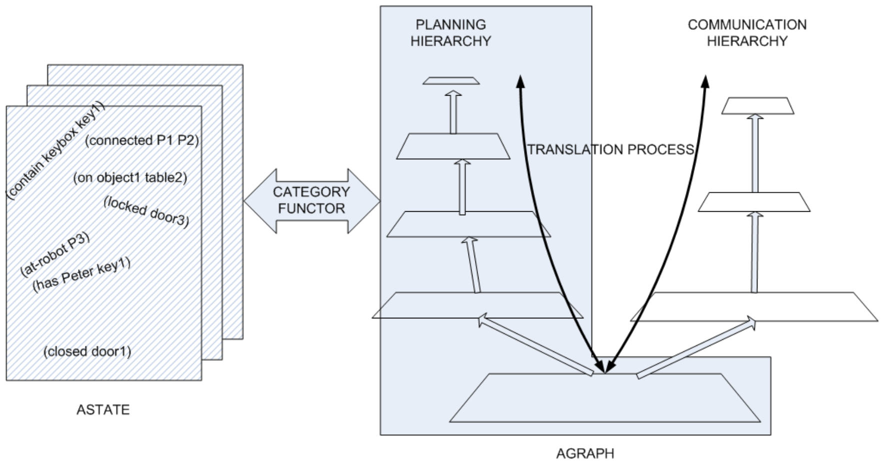

Figure 1. General scheme of our interactive task planning approach based on multiple abstraction. In our multi-hierarchical model, the planning hierarchy (the shadowed one) is devoted to efficient planning, while the other is aimed to support symbolic human-machine communication. Our formalization also permits us to relate classical planning (left box) to symbolic abstraction.

Our symbolic, graph-based representation of the spatial environment includes multiple abstraction and: i) has demonstrated its suitability to use the anchoring paradigm (see [41]), ii) is able to deal with the combinatorial explosion of classical planning processes (demonstrated previously in [6]), and iii) is more efficient than using one symbolic representation both for planning and for human-machine interaction. Actually, we use one hierarchy for improving efficiency of a classical task planning algorithm, and a second one for communicating with the human also efficiently in the sense that it contains the best symbols for human understanding (see fig. 1 for an overall scheme of our approach). In the case that more than one person uses the application, several hierarchies for communication could be added. The planning hierarchy should be automatically constructed for improving the efficiency of planning (as has been reported previously in [7] and [25]). The communication hierarchy, on the other hand, is constructed under the preferences of a human.

The rest of the paper is structured as follows: section II presents a formalization of our symbolic graph-based, multi-hierarchical model of the environment where planning is to be done; section III provides a formalization of human-robot task

[1] The human cognitive map is the body of knowledge about the physical environment that is acquired and used, generally without concentrated effort, to find and follow routes from one place to another, and to store and use the relative positions of places ([18],[19]).

planning interaction in that context and an example of its possibilities; section IV illustrates with a real experience the use of our approach. Finally, we outline some conclusions and future work.

II. A Multi-Hierarchical Symbolic Model

We have chosen an explicit, symbolic representation of knowledge through annotated graphs ([7]) which allows us to manage any type of symbols and relations between them and also to include non-relational information in the form of annotations. This representation has a direct utility, for example, in topological modeling of spatial environments, but here we also use it to model objects (not only topological places) and relations different from spatial reachability. We have then enhanced the annotated graphs with a multi-hierarchical structure, obtaining the so called *Multi-AH-graph*, that is able to maintain several interconnected views of the machine workspace: some views are good for efficient planning and others for human-machine communication.

In our previous works, the multi-hierarchical symbolic model, which was initially used only for robot path-search in [7], has been exploited for different aims. In [6] a single-hierarchical model is used for performing efficient task planning, without user participation into the process (the user only provides the goal of planning). The work presented in [25] copes with the automatic construction of the planning hierarchy of the model for minimizing certain objective functions (like the cost of planning for the most common tasks), but, once again, without user participation in the planning process. Finally, [5] has reported a robotic control architecture featured with a single-hierarchical symbolic model of the space that permits a mobile robot to perform within large environments considering human participation in the plan execution but not during the planning process. The work that we present here copes with the user participation during the planning process. For that we consider the general multi-hierarchical model presented in [7] but instantiated with two hierarchies, each one devoted to solve a different problem (planning and user communication/interaction, respectively).

Subsection II-A presents a formalization of Multi-AH-Graphs using Category Theory ([16]). This permits us to formally relate the graph-based representation of the environment where plans are executed to the clause-based representation of actions and states of the world needed by standard AI planners, i.e. STRIPS, Metric-FF, etc. ([14],[15]). Next, we propose an instantiation of our model with two hierarchies, the *Planning Hierarchy* (subsection II-B), for boosting the planning process, and the *Communication Hierarchy* (subsection II-C), to fit the human communication needs and preferences.

A. Formalization of Multi-Hierarchical Graphs

A Multi-AH-graph is a graph-based, symbolic representation of real environments that includes multi-hierarchical information, that is, the possibility of abstracting groups of elements to super-elements in different and simultaneous ways. From a constructivist point of view, we use a kind of abstraction that produces different layers (flat graphs) isolated from one another, called *hierarchical levels*, which represent the same environment with different amounts of detail. Vertexes of each hierarchical level represent elements of the world (e.g. places, objects, etc.) possibly annotated with non-relational information (shapes, colors, etc.), while edges represent relations between them, possibly holding weights to indicate their strength. For example, vertexes can represent distinctive places for robot navigation, while a set of edges indicates the navigability relation between them with geometric distances as weights. The lowest hierarchical level is called the *ground level*: it represents the world with the maximum amount of detail available.

Formalization of Abstraction of Graphs. In order to propose a formalization of Multi-AH-Graphs, we first need to formalize abstraction of graphs.

Given two non-empty, finite, directed multigraphs without loops *G* and *H*, each defined as a tuple (following [20]):

$$(V, E, \gamma, ini, ter)$$

where *V* is a finite set of *vertexes*, *E* a finite set of *edges*, γ the *incidence function*, *ini* the *initial function*, and *ter* the *terminal function*, an *abstraction from graph G to graph H* can be defined as:

$$A = (G, H, \nu, \varepsilon)$$

where *G* is the graph that is abstracted, *H* is the resulting graph, ν is the *abstraction function for vertexes*, and ε is the *abstraction function for edges*. The following restrictions must hold[2]:

$$\nu : V^G \to V^H \textit{ is a partial function}$$

$$\varepsilon : E^G \to E^H \textit{ is a partial function}$$

Such that[3],

[2] In the following we denote with a superscript the graph to which each component of the abstraction belongs. That is, V^G represents the set of vertexes of graph *G*.
[3] Given a function f(x), def(f(x)) holds iif f(x) is defined, i.e. there exists "y": f(x)=y.

$$\forall z \in E^G, def(\varepsilon(z)) \Rightarrow [def(\nu(ini^G(z))) \wedge def(\nu(ter^G(z)))]$$
$$\forall z \in E^G, def(\varepsilon(z)) \Rightarrow [\nu(ini^G(z)) \neq \nu(ter^G(z))]$$
$$\forall z \in E^G, def(\varepsilon(z)) \Rightarrow \begin{bmatrix} \nu(ini^G(z)) = ini^H(\varepsilon(z)) \wedge \\ \nu(ter^G(z)) = ter^H(\varepsilon(z)) \end{bmatrix} \tag{1}$$

That is, an edge can be abstracted if and only if its initial and terminal vertexes have been abstracted into two different supervertexes (see figure 2). The vertex $\nu(a)$ for a given vertex $a \in V^{(G)}$ is called the *supervertex* of *a*. Analogously, the edge $\varepsilon(z)$ for a given edge $z \in E^{(G)}$ is called the *superedge* of z.

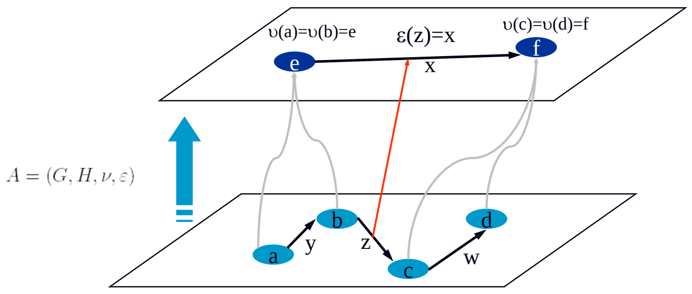

Figure 2. An example of abstraction of graphs. Notice that according to the restrictions imposed in (1), edges *y* and *w* can not be abstracted.

In the case that both υ and ε are total (every element of their domains having an image), we will call the whole abstraction *complete*. In the case that both υ and ε are on-to (every element of their ranges having a defined correspondence with an element in their domains), the whole abstraction will be called *covered*.

Functions ν and ε have inverses defined as:

$$\nu^{-1}: V^H \to power(V^G) \qquad \varepsilon^{-1}: E^H \to power(E^G)$$
$$\forall b \in V^H, \nu^{-1}(b) = \{a \in V^G : \nu(a) = b\} \qquad \forall y \in E^H, \varepsilon^{-1}(y) = \{z \in E^G : \varepsilon(z) = y\}$$

where *power(C)* denotes the set of all the subsets of *C*. These inverse functions are called *refining functions* for vertexes and edges, respectively. For any vertex $a \in V^H$, the vertexes belonging to $\nu^{-1}(a)$, if any, are called the *subvertexes* of *a* in *G*. Analogously, for any edge $z \in E^{(H)}$, the edges belonging to $\varepsilon^{-1}(z)$, if any, are called the *subedges* of z in *G*.

Formalization of Multi-AH-Graphs through Category Theory. As it will be explained further on, Multi-AH-Graphs are mathematically finite subsets of the category of graphs with abstractions, called here *AGraph*, which is similar to the well-known Category *Graph* of graphs with homomorphisms [21], except that it is defined under the above specification of graph abstraction, that is a partial morphism.

Thus, the *AGraph* category can be formally specified by:

$$AGraph = (\Theta, \nabla, \ell^-, \ell^+, I, \bullet)$$

where Θ is the collection of all possible non-empty, finite, directed multigraphs without loops [20], ∇ is the collection of all possible abstractions of graphs following our definition of abstraction, ℓ^- is the *lower hierarchical level* function, ℓ^+ is the *higher hierarchical level* function, *I* is the *identity* function, and $\bullet$ is the *composition of abstractions* function, such that:

$\ell^-: \nabla \to \Theta$ and $\ell^+: \nabla \to \Theta$ are functions that yield the two graphs involved in a given abstraction. That is, for $A = (G, H, \nu, \varepsilon)$, $\ell^-(A) = G$, and $\ell^+(A) = H$.

$I: \Theta \to \nabla$, for any graph G yields an abstraction that leaves it unaltered: $I(G) = (G, G, \nu_G, \varepsilon_G)$, where:

$$\nu_G: V^G \to V^G \qquad \varepsilon_G: E^G \to E^G$$
$$\forall a \in V^G, \nu_G(a) = a \qquad \forall z \in E^G, \varepsilon_G(z) = z$$

Finally, $\bullet: \nabla \times \nabla \to \nabla$ is a partial function that yields the composition of two given abstractions *A1*, *A2* as long as $\ell^+(A_1) = \ell^-(A_2)$ (otherwise it is undefined). It is constructed as follows:

$$A_2 \bullet A_1 = (\ell^-(A_1), \ell^+(A_2), \nu_\circ, \varepsilon_\circ)$$

The two abstraction functions of $\bullet$ are defined by mathematical composition[4]: $\nu_\circ = \nu^{A_2} \circ \nu^{A_1}$ and $\varepsilon_\circ = \varepsilon^{A_2} \circ \varepsilon^{A_1}$. (2)

This composition of abstractions is associative:

$$\begin{aligned} &\forall G, H, J, K \in \Theta, \\ &\forall A_1 = (G, H, \nu_1, \varepsilon_1), A_2 = (H, J, \nu_2, \varepsilon_2), A_3 = (J, K, \nu_3, \varepsilon_3) \in \nabla, \\ &(A_3 \bullet A_2) \bullet A_1 = A_3 \bullet (A_2 \bullet A_1) \end{aligned} \tag{3}$$

Finally,

$$\begin{aligned} &\forall G, H \in \Theta, \forall A = (G, H, \nu, \varepsilon) \in \nabla \\ &A \bullet I(G) = A = I(G) \bullet A \end{aligned} \tag{4}$$

Under our definition for graph abstraction, constraints (2),(3),(4) can be easily demonstrated, and thus *AGraph* is a category.

For the purposes of this paper, we will consider a subcategory of *AGraph* that we will call *CVAGraph**, standing for "*Complete, coVered AGraph with only connected graphs*", that is, a portion of *AGraph* with only complete and covered abstractions on graphs whose vertexes are all connected through some path. For practical interactive task planning we will consider only Multi-AH-Graphs in which all the hierarchies (paths of abstraction) share a common ground hierarchical level. That level will serve as the link between the planning and the communication hierarchies.

B. *The Planning Hierarchy*

Computational efficiency in classical task planners has been largely studied in the AI field, but rarely in the robotics arena. However, in those applications in which a mobile robot performs within a real and large environment, i.e. an office building, classical AI planning often becomes an intractable problem [6]. For these cases, the most adopted solution relies on some heuristic mechanism to simplify the problem at hand. There are some approaches, known as *hierarchical planners*, that use some type of abstraction to speed up planning ([22],[23],[24]). In our work we have employed *Hierarchical Planning through World Abstraction* (HPWA, [6]), which uses abstraction on the description of the world. The HPWA framework is basically a scheme that embeds an existing planner (the so-called *embedded planner*) for doing planning at different levels of a hierarchical representation of the environment, in order to improve planning efficiency in large and complex scenarios. First, a plan is found at a high level of abstraction of the world representation (low detailed). Then, that abstract plan is used to rule out irrelevant information at the next lower level (more detailed level) of the hierarchy, creating a sequence of plans more and more detailed. The performance of HPWA has been demonstrated elsewhere [6], comparing its efficiency to other classical planners when solving tasks in complex and large environments (see figure 3). In general, HPWA yields excellent results when planning in large environments, but its efficiency is largely tight to the particular hierarchical representation (symbolic hierarchy), and thus, a special attention should be paid to the automatic construction of appropriate hierarchies for planning. Although this topic is out of the scope of this paper, it deserves a brief explanation. Constructing the best hierarchy, that is, the best arrangement of symbols for planning, is an intractable problem since it involves constructing all the possible hierarchies that can be constructed upon the set of ground data[5] and evaluating them with respect to the considered tasks. An additional problem is that the hierarchical representation should capture the dynamism of the environment with the possibility of modifying both the ground data (e.g. a new object has been found) and the tasks to be planned (e.g. the robot is commanded a new mission). Due to this complexity, there are not many works in the literature addressing this topic, though it has been explored recently in [25], where an evolutionary algorithm is considered for continuously searching for a good hierarchy for planning under a set of variable robot tasks and also coping with environment dynamics.

[4] The superscript denotes a particular element from a given abstraction, that is, ν^{A_1} indicates the abstraction function for vertexes defined in the abstraction A_1.

[5] The number of the hierarchies that can be constructed upon a set of ground vertexes involves the Bell's number. For a very reduced environment, considering only 10 vertexes, this figure is B(10)=115975.

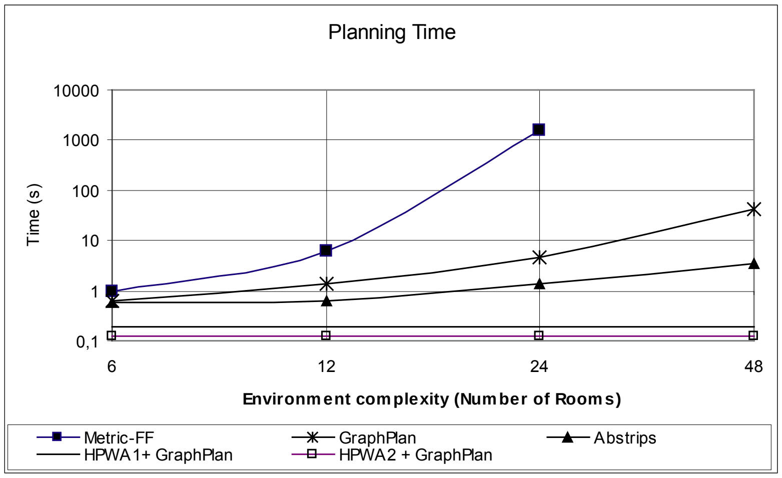

Figure 3. Planning efficiency of HPWA. As long as the planning domain grows, the performance of classical planners largely decreases. The HPWA framework is capable of detecting and maintaining only the information necessary for solving the task at hand, ruling out irrelevant information, and thus, achieving high performance in complex and large environments.

The AState Category and its relation to AGraph. In a nutshell, classical planning consists of searching an ordered sequence of actions, i.e. (GO *origin destination*), that modify the current state of the world (generally represented through sets of logical predicates) to attain a goal state. In our case, the symbolic world information required for planning is stored in our graph-based model, that is, it is an annotated graph, and thus it must exist a close relation between *graphs* and *planning states*. Moreover, HPWA works in a hierarchical fashion, and therefore we must also establish a relation between graph abstraction and abstraction of planning states. All these relations can be formalized by defining a Category for planning states with abstraction (that we call the *AState* Category), and relating it to the *AGraph* category through *functors* ([16]).

A *planning state* (a *state* for short) is a finite and consistent set of logical predicates in the form *p=(predicate_name* $param_1, \ldots param_k$*)* that represents some piece of world information. We denote as $\mho$ the set of all possible planning states over a certain language[6] *L*, $\wp$ as the set of all possible symbol predicates, and *Param* as the set of all possible parameters that can be defined over *L*. Given a planning state $S \in \mho$, we will need a *State Parameters* function, $SP(S)$, a function that yields the distinct parameters from all logical predicates of *S*, and a *State Predicate Names* function, $SN(S)$, that yields the finite set of distinct predicate names of *S*. When an enumeration of elements from *SP(S)* or *SN(S)* are needed, we will use the notation *SN(S)[i]* (*SP(S)[i]*) to refer to the *i-th* element. We also extend the meaning of these functions to be applied to simple predicates instead to states. That is, for the predicate *p=(at book-1 table-1)*, *SN(p)=<at>*, *SP(p)={<book-1>, <table-1>}*, and *SP(p)[2]=<table-1>*.

Similarly to graph abstraction, we can define the abstraction of planning states: an *abstraction* A_s from a planning state *S* to a planning state *T* is a morphism between both states, defined as a tuple:

$$A_s = (S, T, \xi, \pi), \text{ where}$$
$$\xi : SN(S) \rightarrow SN(T) \textit{ is a partial function}$$
$$\pi : SP(S) \rightarrow SP(T) \textit{ is a partial function}$$

Correspondingly to the definition of *AGraph*, we formulate now the Category *AState* of planning states with abstractions. Given a first-order language *L*,

$$AState(L) = (\mho, \blacktriangle, \varpi^-, \varpi^+, I, \bullet)$$

where $\mho$ is the collection of all possible states defined in *L*, $\blacktriangle$ is the collection of all possible abstractions on those states, ϖ^- is the *refined state* function, ϖ^+ is the *abstracted state* function, *I* is the *identity* function, and $\bullet$ is the *composition of state abstractions* function. Definitions are analogous to those given in section II.A.

Once both categories are formalized, we can relate them by means of a *functor*. Funtors are functions that relate objects and arrows of two categories, preserving their structures. We formalize the *Graph-State Functor* $\Psi = (\Psi_o, \Psi_a)$, between the categories *CVAGraph** and *AState*, as follows:

$$\Psi_o : \Theta^* \rightarrow \mho$$

[6] In the rest of the paper, we assume that *AState* is defined on a certain first-order language *L* which will not be explicitly specified any more.

$$\Psi_a : \nabla \rightarrow \blacktriangle$$

being both of them total functions. Informally, Ψ permits us to transform a graph into a planning state through Ψ_o, and an abstraction of graphs into an abstraction of states that preserves the former transformation by means of Ψ_a (see figure 4). The complete formalization of Ψ requires the following auxiliary functions (the reader can safely skip these details and go to subsection II-C) .

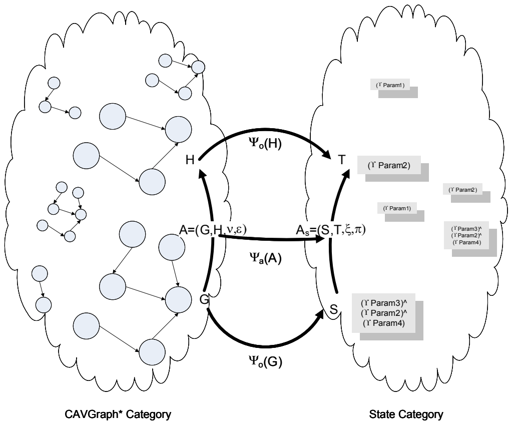

Figure 4. The Graph-State Functor. Functor Ψ maps objects and arrows between the *AGraph* (the world) and *AState* (the planning space) categories. Its inverse is not always defined.

The *Vertex-Param Translator*, Γ_v, is a partial, one-to-one, and on-to function from the set of vertexes of a graph to the set of all possible parameters considered in the planning domain (noted as *Param*). For instance, given a graph *G*, $\Gamma_v(a)$ =<*my desk*>, where $a \in V^{(G)}$, <*my desk*> $\in$*Param.*

The *Edge-Predicate Translator*, Γ_e, is a partial function from the set of edges of a graph to the set of all possible predicate names considered in the planning domain (noted as $\wp$). It is defined as follows:

$$\Gamma_e : E^{\Theta} \rightarrow \wp \times \wp \times \wp$$

$$\forall z \in E^{\Theta}, \Gamma_e(z) = (g, h, i) : g, h, i \in \wp$$

For example, if a certain edge, let say *w*, indicates a navigability relation between two places (vertexes), then $\Gamma_e(w)$ could be defined as $\Gamma_e(w) = (<location>, <location>, <navigable>)$. That is, Γ_e transforms the relational information of edges (the world) into predicates (planning components) that refer to the same information. We also define three functions for retrieving separately each of the predicate names yielded by Γ_e. Thus,

$$\forall z \in E^{(\Theta)}, \Gamma_e(z) = (g, h, i) \Leftrightarrow \Gamma_{e1}(z) = g \wedge \Gamma_{e2}(z) = h \wedge \Gamma_{e3}(z) = i$$

The *Edge-State Translator*, β_G is then defined as a total function based on the definition of Γ_e and Γ_v, that yields a set of logical predicates that represent the information represented by the edges in a given graph *G* and their related vertexes:

$$\beta_G : E^G \rightarrow \mho$$

$$\forall z \in E^G, \beta_G(z) = \left\{ \begin{matrix} p_1 = \left(\Gamma_{e1}(z) \quad \Gamma_v(ini(z))\right) \\ p_2 = \left(\Gamma_{e2}(z) \quad \Gamma_v(ter(z))\right) \\ p_3 = \left(\Gamma_{e3}(z) \quad \Gamma_v(ini(z)) \quad \Gamma_v(ter(z))\right) \end{matrix} \right\}$$

Informally, given a graph *G*, β_G transforms edges (the world) into three logical predicates (planning components) that express information regarding the vertexes involved in edges as well as about the type of relation between them For instance, for the navigability edge *w* commented before, that connects two vertexes *a* and *b*, such that $\Gamma_v(a)$ =<*my desk*> and $\Gamma_v(a)$ =<*my office's door*>, the edge-state translator yields the following set of predicates:

$\beta_G(w) = \{$ `(location <my desk>),(location <my office's door>),(navigable <my desk> <my offices's door>)` $\}$

Finally, through the use of β_G, the Ψ_o functor can be directly defined as:

$$\Psi_o : \Theta \to \mho$$
$$\forall G \in \Theta, \Psi_o(G) = \bigcup_{z \in E^G} \beta_G(z)$$

The second component of Ψ, the functor for arrows Ψ_a, permits us to transform abstractions of graphs into state abstractions, preserving the original structure. It can be defined as:

$$\Psi_a : \nabla \to \blacktriangle$$
$$\forall A \in \nabla, \Psi_a(A) = A_s = (\Psi_o(G^A), \Psi_o(H^A), \tau, \kappa)$$

where functions τ and κ are defined as:

$$\tau^{A_s} : \wp \to \wp$$
$$\forall z \in E^{(G^A)}, \begin{bmatrix} \tau^{A_s}(\Gamma_{e1}(z)) = \Gamma_{e1}(\varepsilon(z)) \\ \tau^{A_s}(\Gamma_{e2}(z)) = \Gamma_{e2}(\varepsilon(z)) \\ \tau^{A_s}(\Gamma_{e3}(z)) = \Gamma_{e3}(\varepsilon(z)) \end{bmatrix}$$

$$\kappa^{A_s} : Param \to Param$$
$$\forall a \in V^{G^A} : \kappa^{A_s}(\Gamma_v(a)) = \Gamma_v(\nu(a))$$

Through functions τ and κ the logical predicates that make up a planning state can be abstracted following a given graph abstraction. That is, it is the medium by which the planning process can utilize the hierarchical information stored in the Multi-AH-Graph.

C. The Communication Hierarchy

Psychology has revealed that humans seem to group symbols at different levels of detail in order to manage efficiently large amounts of information ([53],[54],[55]). For instance, when thinking about an office building, we rapidly sketch a hierarchical structure distinguishing groups of floors, sets of offices and corridors inside each floor, furniture at each office, etc. However, different individuals (even within the same environment) may exhibit particular preferences to group information and set different labels for identifying each group. For example, a cleaner may group a set of offices that share their dirtiness attaching them the label "hard work area", which is not done by a visitor. The same is obviously valid when different languages are used.

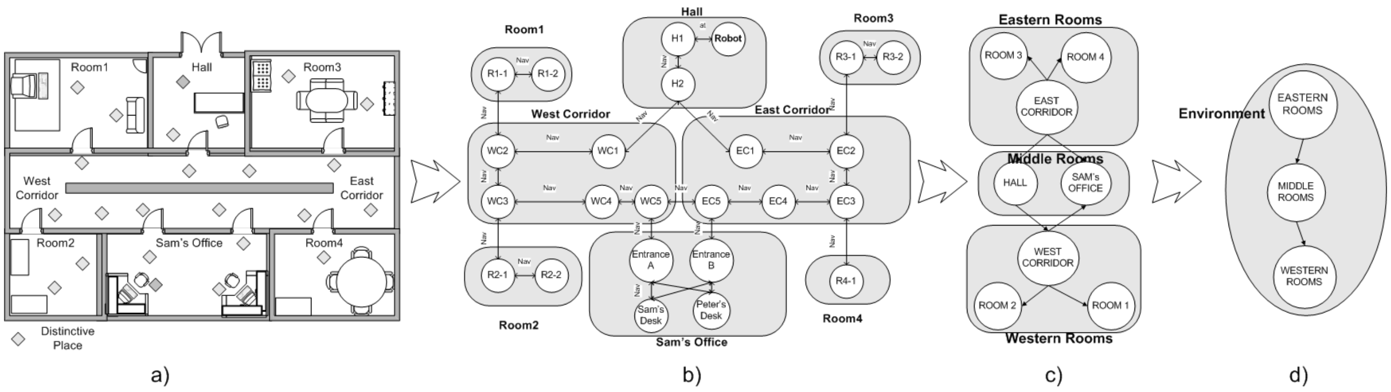

Figure 5. Hierarchical representation of an office environment. (a) The environment. (b) Ground level: topology of distinctive places. (b)-(d) A communication hierarchy for a particular user.

Within our model, the *communication hierarchy* is aimed to arrange world information in this human-like manner. As will be explained further on, through this hierarchy the user can interact with a robot using her/his own set of symbols and assigned labels. Since particular users may have different ways to model the same environment, her/his participation is needed in the construction of the hierarchy.

In our work we rely on an interactive construction process in which the user guides the robot to a particular place, i.e. the entrance door to an office, notifying that a symbol has to be created into the model with a certain label, i.e. "the door of my workplace". In this way, the user can provide the robot with a set of ground symbols, which are vertexes at the ground level of our Multi-AH-Graph and represent distinctive places or physical objects, and also can include relations between them, like connectivity or navigability, modeled through edges[7]. Subsequently, the user can select a set of vertexes to make up abstract symbols like rooms or areas. Figure 5 shows an example of the communication hierarchy constructed in this way for a typical office environment.

III. Human-Machine Interactive Task Planning

Our interactive task planning process exploits the multi-hierarchical model presented in section II. It permits the user to command a machine to solve a task through her/his own set of symbols (taken from the communication hierarchy), while the machine can efficiently plan the task using the planning hierarchy and communicate the results using concepts understandable by the user (via a translation of symbols from the planning hierarchy to the communication one). Moreover, when partial planning results are available, as in the HPWA case, the user can be reported with the evolution of the process, enabling her/him to guide the planning algorithm and propose modifications based on commonsense knowledge or particular preferences. Such an interactiveness can be deactivated by the user at any moment, making the hierarchical planner to provide a final plan for the task at hand which is reported to the human by means of ground symbols.

Subsection III.A deals with the formalization of our interactive task planning process. Subsection III.B illustrates all the modalities that our interactive process permits to the user.

A. Formalization of the Interactive Task Planning Process

Our interactive task planning approach uses iteratively these phases:

1) Translation of the user's requested goal, which is specified using symbols of some level of the *communication hierarchy*, into the *planning hierarchy*. This is done by refining the symbols involved in the goal until reaching the common ground level of the Multi-AH-Graph. For example, if the user goal is "go to *my workplace*", the human symbol (*my workplace*) involved in the task is moved down to the ground level, choosing as refined symbol one of their subvertexes, let say *my desk*. Thus, the user goal turns into "go to *my desk*".

2) Abstracting the goal through the planning hierarchy for finding a first sketch of a plan at a high level of abstraction (low detail). Usually, this is done at the highest levels of the planning hierarchy for improving efficiency.

3) Reporting the abstract plan to the user, translating it to the communication hierarchy. The human decision about that plan (to reject or accept it, total or partially) may involve a backwards translation to the planning hierarchy and a subsequent step of planning.

In stages 1) and 3) it is necessary to pass symbolic information from one hierarchy to the other, particularly plans. Informally, refining and abstracting plans within a hierarchy consists of refining/abstracting the sequence of actions involved in the plan. A *plan action* is the instantiation, with variables of the domain, of a plan operator composed of a pair of logical predicates <*Precond*,*Postcond*>. If at a certain moment the logical predicates of *Precond* are satisfied, the action can be executed and the environment modified according to *Postcond* by adding or eliminating information[8].

Abstracting a plan produces a more general one, since it contains more abstract symbols. Conversely, refining a plan yields a set of more detailed plans covering all possible combinations of the refinements for the parameters of the plan. Both mechanisms can be formalized through the *Plan Abstraction* and the *Plan Refinement* functions defined as follows.

The *Plan Abstraction Function* (in short, *PlanUp*) serves to abstract plans that have been produced at a certain level of the planning hierarchy. For clarity, to define the *PlanUp* function, we firstly define the *action abstraction function* (*ActionUp*).

ActionUp is a partial function that abstracts actions of a plan, like (*GO r1 n1 n2*), if and only if their pre- and postconditions can be abstracted. Formally, given an abstraction in *CVAGraph**, $A=(G,H,\nu,\varepsilon)$ and its corresponding abstraction in *AState* $A_s=(Q,S,\xi,\pi)$ (constructed by the Ψ_a functor), let Σ_Q (Σ_S) be the set of all possible actions involving parameters only from the planning state *Q* (*S*). The *ActionUp* function is defined for an action *a* as:

$$ActionUp_{A_s} : \Sigma_Q \to \Sigma_S \text{ is a partial function}$$
$$\forall a \in \Sigma_Q, def(ActionUp_{A_s}(a)) \Leftrightarrow \forall p \in (a^{Precond} \cup a^{Postcond}), def(\xi(SN(p)) \wedge$$
$$\forall j \in 1..|SP(p)|, def(\pi(SP(p)[j]))$$

When defined,

[7] In the case of a robotic application, non-structural data is automatically attached to vertexes and edges in the form of annotations, like robot pose, camera images, etc., for performing the operations that are planned (navigation, manipulation, etc.).

[8] Plan actions given in the form (action-name $param_1$,…$param_i$) can be considered as logical predicates. Similarly, complete plans, as sequence of actions, can be considered planning states, and thus objects of the *AState* Category.

$$ActionUp_{A_s}(a) = a' \in \Sigma_S :$$
$$\begin{bmatrix} |SP(a')| = |SP(a)| \wedge \\ SN(a') = SN(a) \wedge \\ \forall j \in 1..|SP(a')|, SP(a')[j] = \pi(SP(a)[j]) \end{bmatrix}$$

The *Plan Abstraction Function*, *PlanUp*, can then be formalized. Given an abstraction in *CVAGraph**, let say $A = (G, H, \nu, \varepsilon)$, and its corresponding abstraction in *AState*, $A_s = (Q, S, \xi, \pi)$, let $\mho_Q$ ($\mho_S$) be the set of all possible plans whose actions are in Σ_Q (Σ_S). *PlanUp* is defined as:

$$PlanUp_{A_s} : \mho_Q \to \mho_S$$
$$\forall p = (a_1, a_2, \ldots, a_n) \in \mho_Q,$$
$$PlanUp_{A_s}(p) = p' = (a'_1, a'_2, \ldots, a'_n) \in \mho_S :$$
$$\forall a'_i \in p', a'_i = ActionUp_{A_s}(a_i)$$

On the other hand, the *Plan Refinement Function* (in short, *PlanDown*) yields a refined version of a plan *p* based on refinement of graphs. As before, we first define the *action refinement function* (*ActionDown*) and then the equivalent definition for plans.

Given an abstraction in *CVAGraph**, $A = (G, H, \nu, \varepsilon)$ and its corresponding abstraction in *AState*, $A_s = (Q, S, \xi, \pi)$ (constructed by the Ψ_a functor), let Σ_Q (Σ_S) be the set of all possible actions involving parameters from *Q* (S). The *ActionDown* function is always defined[9] and when it is applied to an action *a* yields a set of actions {*a'*} such that:

$$ActionDown_{A_s} : \Sigma_S \to power(\Sigma_Q)$$
$$\forall a \in \Sigma_S,$$
$$\begin{bmatrix} \forall a' \in ActionDown_{A_s}(a), \\ \big[|SP(a')| = |SP(a)|\big] \wedge \big[SN(a') = SN(a)\big] \wedge \\ \Big[\forall j \in 1..|SP(a')|, SP(a')[j] \in \big[\pi^{-1}\big]^{A_s}(SP(a)[j])\Big] \wedge \\ \Big[\big(\exists x1, x2, \in 1..|SP(a)| : SP(a)[x1] = SP(a)[x2]\big) \Rightarrow SP(a')[x1] = SP(a')[x2]\Big] \wedge \\ \left[\bigcup_{a' \in ARF_{A_s}(a)} SP(a') = \bigcup \big[\pi^{-1}\big]^{A_s}(SP(a))\right] \wedge \\ \Big[\forall j \in 1..|SP(a)|, \forall b \in \big[\pi^{-1}\big]^{A_s}(SP(a)[j]), \exists a' \in ARF_{A_s}(a) : b \in SP(a')\Big] \end{bmatrix}$$

Informally, through this definition we establish that given an action *a*, it is refined on a set of actions that all have the same length and action name. Also we impose that all possible combinations of refined actions are considered, and that several instances of a parameter in *a* must be refined to the same parameter in each refined action.

The *Plan Refinement Function*, *PlanDown*, can now be formalized. Given an abstraction in *CVAGraph**, $A = (G, H, \nu, \varepsilon)$ and its corresponding abstraction in *AState* $A_s = (Q, S, \xi, \pi)$, let $\mho_Q$ ($\mho_S$) be the set of all possible plans whose actions are in Σ_Q (Σ_S). The PRF is defined as:

$$PlanDown_{A_s} : \mho_S \to power(\mho_Q)$$
$$\forall p, (a_1, a_2, \ldots, a_n) \in \mho_S, PRF_{A_s}(p) = \{p'\}$$
$$with\ p' = (a'_1, a'_2, \ldots, a'_n) \in \mho_Q :$$
$$\Big[\forall a_i^s \in p, \forall a^* \in ActionDown_{A_s}(a_i^s), \exists p' \in PlanDown_{A_s}(p) : a_i^{s'} = a^*\Big] \wedge$$
$$\begin{bmatrix} \forall a'_u, a'_w \in p', \\ \begin{pmatrix} \exists x1 \in 1..|SP(a'_u)| \wedge \exists x2 \in 1..|SP(a'_w)| : \\ \pi^{A_s}(SP(a'_u)[x1]) = \pi^{A_s}(SP(a'_w)[x2]) \end{pmatrix} \Rightarrow SP(a'_u)[x1] = SP(a'_w)[x2] \end{bmatrix}$$

[9] We assume in this paper that the refinement of actions is a total function: abstract symbols without subsymbols are not allowed (this is satisfied automatically if using *CVAGraph**).

B. A Sample Scenario for a Human-Robot Interactive Task Planning

Figure 5 depicts a scheme of part of a typical office environment. Upon the ground level the communication hierarchy enables the robot to manage human symbols. Figure 7 shows a multi-hierarchical representation of that environment in which the planning hierarchy arranges properly the world elements with the goal of improving the task planning process.

In this scenario, let us consider the following application. An employee, at the entrance of the office building, is in charge of receiving and distributing mails to the rest of employees. To facilitate his work, a servant robot can carry objects within the office building, so she has only to give the proper envelope to the robot and select the destination, i.e. "go to *Sam's office*".

As commented before, the first stage of the translation process consists of shifting the human concepts involved in the requested goal into concepts of the ground level of the Multi-AH-Graph. This is solved by simply choosing any subsymbol (subvertex), for example *Entrance to Sam's office*. This selection may lead to the user insatisfaction (who probably would prefer *Sam's desk* to *Entrance to Sam's office* as the destination) or to an unreacheable goal. Two possibilities are available then: choose another subvertex or ask the user for a more detailed specification for the goal.

Once an adequate specification of the goal is obtained at the ground level, the hierarchical planner solves the task using the planning hierarchy. HPWA first abstracts all the ground elements to the highest level of the planning hierarchy, and then produces successively plans at different levels, which involves symbols not understandable by the user. Following our example, the planner produces the first abstract plan at level *L3* from the planning hierarchy[10] which is *(GO C1 C2)*. This must be translated into human symbols, thus it must be first refined to the ground level. The first refinement translates it to level *L2* of the planning hierarchy, yielding the list of plans:

$$PlanDown(\{(\text{GO } c1\ c2)\}) = \left\{ \begin{matrix} \{(\text{GO } b1\ b3)\}, \{(\text{GO } b1\ b4)\}, \\ \{(\text{GO } b1\ b5)\}, \{(\text{GO } b2\ b3)\}, \\ \{(\text{GO } b2\ b4)\}, \{(\text{GO } b2\ b5)\} \end{matrix} \right\}$$

that can be simplified as a *plan schema*:

$$\{\ (\text{GO } \{b1,b2\}\ \{b3,b4,b5\})\ \}$$

By successively refining the abstract plans down through the planning hierarchy, a set of plans that only involve parameters which represent distinctive places are generated. Such plans are then abstracted up through the communication hierarchy via the *PlanUp* function. In this example, this yields the set of plans in the communication hierarchy shown in figure 6.

Level 1	{(GO {Hall, East-Corr., West-Corr., Room3} {Sam's Office, East and West Corr., Rooms 1,2,4})}
Level 2	{(GO {West, Middle, East} {West, Middle, East})}
Level 3	N/A

Figure 6. Possible translations of the parameters of an abstract plan from L3 of the planning hierarchy.

Through these translated plans, the user can proceed in the following ways:

a) Inquiring a more detailed plan. The translation of an abstract plan may not provide enough information to the user. In these cases the user can request more information in two different ways: she can ask the robot for a translation of the same plan using more detailed symbols from the communication hierarchy (however, this increases the plan ambiguity), or she can ask the planning process for planning a new solution at a lower level of the planning hierarchy. In this latter case the obtained plan will involve more detailed concepts, which in turn can reduce ambiguity.

b) Rejecting part of a plan. Observe that even when the provided plan does not reveal enough information, the user can interact productively with the planning process, i.e. by rejecting certain spatial symbols. In our example, the user may require the robot to avoid the *West-Corridor* and *Room2* regions for personal preferences or commonsense knowledge of the environment, since they are, for example, crowded that day. Such discarded symbols are translated again into the planning hierarchy, reporting to the hierarchical planner that symbols *b4* at level *L2* must be discarded. Thus, HPWA plans now at level *L2* of the planning hierarchy without considering such a symbol, producing the new plan {(GO *b1 b2*), (GO *b2 b3*)}, which is reported to the user as:

{(GO {Hall, East-Corridor} {East-Corridor, Room3}, (GO {East-Corridor, Room3} {East-Corridor, Sam's Office})}.

[10] The abstract plan at the highest level is not considered in this example since it is trivial (there is a single node).

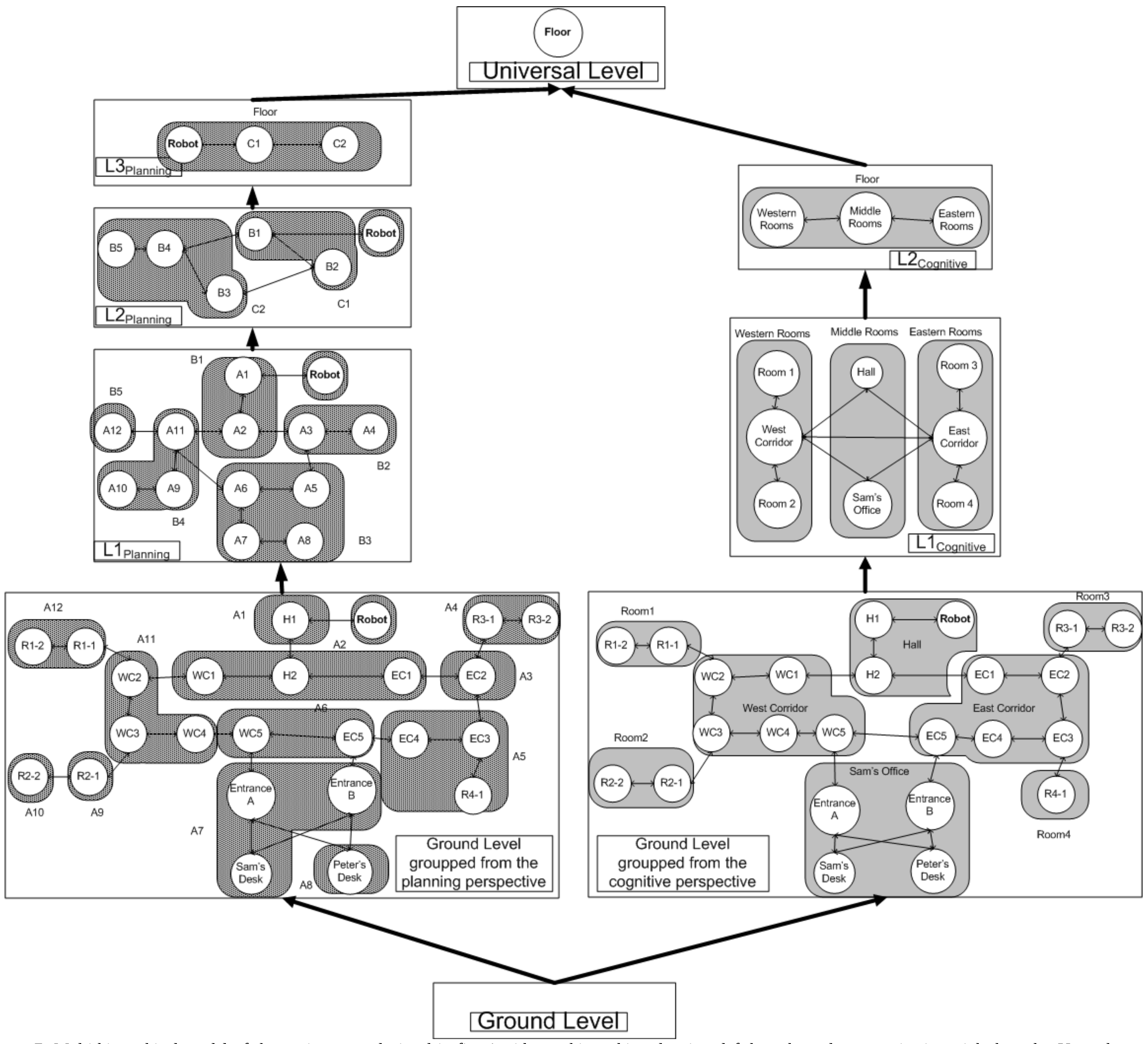

Figure 7. Multi-hierarchical model of the environment depicted in fig. 4 with two hierarchies planning -left branch- and communication -right branch-. Upon the ground level, shade regions and labels indicate the grouping of vertexes into supervertexes. At the ground level, vertexes hold identifying labels that for clarity sake have been set to short codes.

c) Suggesting an abstract plan. Due to the ambiguity involved in the translation process, the user may be informed about a set of different possibilities to solve a plan that maybe improves efficiency of planning. She can select one out of the offered solutions based on her knowledge of the environment. For instance, in our example the user can suggest the abstract plan: {(GO Hall East-Corridor), (GO East-Corridor Sam's Office)} since she knows that it is not necessary to consider *Room3* to arrive to the destination.

Thus, through the solution pointed out by the human, the planner can solve the task considering only those symbols embraced by the ones suggested by her. In this example, the final plan at the ground level is {(GO h1 h2) (GO h2 ec1) (GO ec1 ec2) (GO ec2 ec3) (GO ec3 ec4) (GO ec4 ec5) (GO ec5 l2)}, that would be communicated to the user with the labels given by her, for example:

```
{(GO "Hall's desk" to "Hall's door")
(GO "Hall's door" to "Hall connection to the East Corridor")
(GO "Hall connection to the East Corridor" to "Room3's door")
(GO "Room3's door" to "Corner of the East Corridor")
(GO  "Corner of the East Corridor" to "Room4's door")
(GO "Room4's door" to "Sam's Office door B")
(GO "Sam's Office door B" to "Sam's Office Entrance B")}
```

IV. Real Experiences

The effectiveness achieved by any system in which humans are involved is normally difficult to measure. This is the case of our human-computer interaction mechanism for which the user satisfaction should be somehow evaluated. The main concern of such an evaluation is that it largely depends on the user characteristics (i.e. cognitive abilities, age, gender, etc.) and thus a representative sample of people should be selected, evaluating their opinion from a psychological and cognitive point of view. This evaluation is out of the scope of this paper, thus we rely on the study of a general case of interactive planning within a robotic application to show the suitability of our approach.

The robotic application we consider here entails a real assistant robot (a robotic wheelchair) that provides mobility to physically impaired people within a large-scale building. The user is assumed to have enough cognitive abilities to take decisions and interact with the planning system of the wheelchair as well as manual capabilities to open/unlock doors or manipulate objects. Our experiences have been conducted with the robotic wheelchair SENA (see figure 8), which includes a number of sensors for performing autonomous navigation and map building ([26]).

The successful performance of SENA is supported by the control architecture *ACHRIN* ([5]). One of the main characteristics of this architecture is that it takes into account human abilities as part of the abilities of the robot, e.g. people can participate in the execution. This permits the system to ask the user for help when execution failures occurs, as is the case of a locked door[11]. Through our interactive planning approach, the user can also participate during the planning phase, guiding the system to fulfill her/his requirements. User interaction with the planning system, and also with the rest of the robotic systems, mainly relies on verbal communication based on commercial voice recognition and TTS tools. However, for dealing with large environments voice interfaces may become hard to use due to its inability to properly recognize the user utterances, being necessary to repeat continuously the commands. Thus, the user can also use graphical interfaces to control and interact with the robot.

In our experience, we do not consider re-planning due to execution failures (not solvable neither by the robot nor by the user). The planning system of ACHRIN is HPWA with FF-Metric as embedded planner, and they have been implemented on C++ and improved with additional functions to cope with the interactive planning approach proposed in this work.

Figure 9 (top) shows a plan map of the considered scenario, constructed by a SLAM method ([42]), which is part of the Computer Science building at the University of Málaga, made of four wings connected through a long corridor and a patio. In our experiences we focus on two of those wings with the particular setup shown in figure 9 (bottom, zoomed).

Room doors can be opened, closed or locked. The actions that the user can perform are: *open*, *unlock* a door, *search* for a key in a key box, *take* a key, and *ask anybody for* a key, while the wheelchair only provides mobility (including obstacle avoidance, path and plan planning, etc.). Conventional (non-hierarchical) planning in this type of scenario with hundreds of distinctive places and objects becomes a complex, even intractable, problem, as has been demonstrated in [6].

[11] You can visit http://www.youtube.com/watch?v=D2oLBzlEEWA to watch one of our videos where SENA performs within an office scenario overcoming different situations like a closed door, navigation errors, etc. thanks to the user help. This video does not show the interactive planning but only the execution of a final plan.

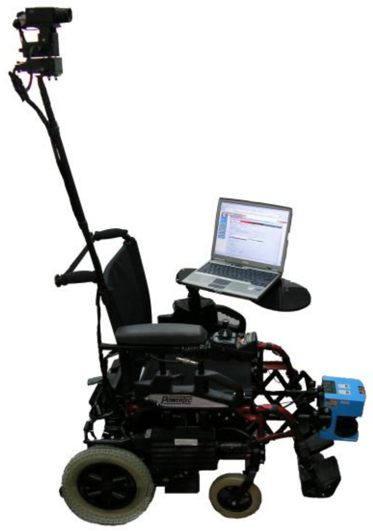

Figure 8. The robotic wheelchair SENA. It is based on a commercial powered wheelchair which has been endowed with several sensors and devices. The SW architecture runs on the user's laptop, who can still use it for her/his work.

Figure 9 (bottom) shows a scheme of the scenario with the vertexes (symbols) of the first level of the communication hierarchy involved in our experiences. We have considered an experience in which the wheelchair user is at his office (*room-6*), and wants to go to *room-9*, that is locked. Thus, the user commands the wheelchair at the first level of the communication hierarchy: "GO to *Room-9*", being necessary its refinement downto the ground level (partially shown in figure 10). An example of such a refinement would be "GO to *Desk*" (inside Room-9), which becomes the ground goal to be solved through the planning hierarchy.

The first solution to the proposed goal is communicated to the user, after the proper translation between hierarchies through the *PlanDown* and *PlanUp* functions, using symbols of the first level of the communication hierarchy (the same that the user employed in his initial command) as: "GO to *Corridor-2",* "GO to *Room-7*", "TAKE a *Key* from the *KeyBox*", "GO to *Corridor-2"* "GO to *Room-9*". Although this is the shortest and easier way to get to the destination, the user may prefer another solution which enables him to mix with his workmates. Thus, he asks the system for an alternative.

The next possible solution in this scenario is abstracted to the first level of the communication hierarchy through *PlanUp* as "GO to *Corridor-2",* "GO to *Corridor-1",* "GO to *Room-1*", "ASK *Peter* for a *Key*", "GO to *Corridor-1",* "GO to *Corridor-2",* "GO to *Room-9*", that although requires more execution time, it would seem more acceptable for the user. Since the user knows that Peter is normally busy, he decides to phone him before going to his office in order to be sure he has not left. Peter informs the user he is going out to a meeting, so the user also rejects this plan since he knows that Peter will not be able to give him the key (this is something that the planning process was unable to foresee).

The planner then goes for another solution, yielding the plan: "GO to *Corridor-2",* "GO to *Corridor-1",* "GO to *Corridor-3",* "GO to {*Room-11*,*Room-12*}" "ASK {*Mark*,*Mary*} for a *Key*", "GO to *Corridor-4",* "GO to *Corridor-2",* "GO to *Room-9*". Notice that there are four possibilities embedded in this plan: (*Room-11*, *Room-12*) x (*Mark*, *Mary*). Any of these should be feasible (the user knows that both Mark and Mary are always at their offices), but the user still has certain preferences, for instance, he has strongly argued with Mark the same day, so he prefers not to meet him, choosing the second option: ask Mary for the keys.

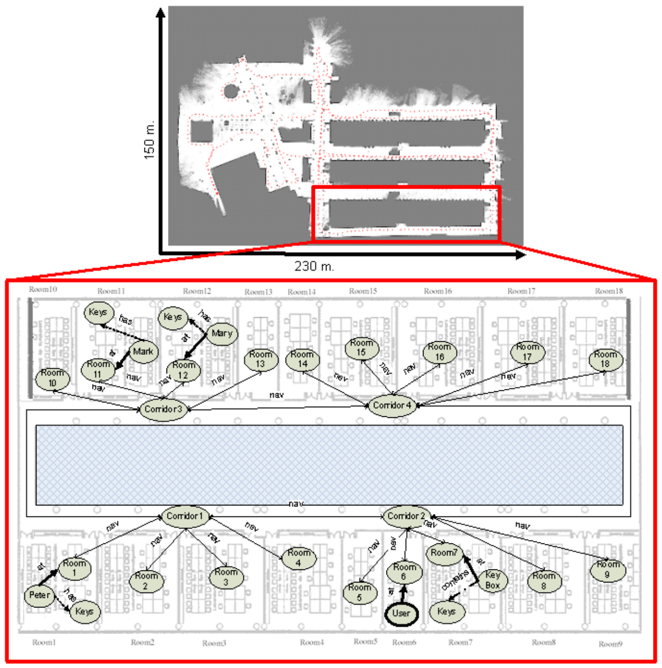

Figure 9. Real scenario where the experiences of interactive task planning have been conducted. (Top) The whole environment entails four wings, a long corridor and a patio. Small dots represent distinctive places. (Bottom) Part of the scenario where only the symbolic information of the first (not ground) level of the communication hierarchy is shown. Different line styles indicate different types of edges.

Once this plan at the first level of the communication hierarchy is selected, the planning process refines it to the next lower level, in this case:

"GO out through *door of office 6"*, "GO to *entrance-office 5*", "GO to *entrance-office 4*", "GO to *entrance-office 3*", "GO to *entrance-office 2*", "GO to *entrance-office 1*", "GO to *entrance-office 10*", "GO to *entrance-office 11*", "GO to *entrance-office 12*", "GO in through *door of office 12*" "ASK *Mary* for *key 1*", "GO out through *door of office 12*", "GO to *entrance-office 13*", "GO to *entrance-office14*", "GO to *entrance-office 15*", "GO to *entrance-office 16*", "GO to *entrance-office 17*", "GO to *entrance-office 18*", "GO to *entrance-office 9*", "UNLOCK *door of office 9*" "GO in through *door of office 9*" "GO to *Desk*".

In this situation, although the high-level plan of asking Mary for the key is accepted, the refined plan yielded by *PlanDown* involves passing close to the Mark's office, which may also be rejected by the user, who does not want to meet him at the corridor. So, he commands to reject the symbol "*Door-11*", and the alternative path for this task is planned and reported to the user as[12]: "GO out through *door of office 6"*, "GO to *entrance-offices 7, 8, 9, 18, 17,16,15,14, 13, 12*", "GO in through *door of office 12*" "ASK *Mary* for *key1*", "GO out through *door of office 12*", "GO to *entrance-offices 13, 14, 15, 16, 17, 18, 9"*, "UNLOCK *door of office 9*" "GO in through *door of office 9*" "GO to *Desk*".

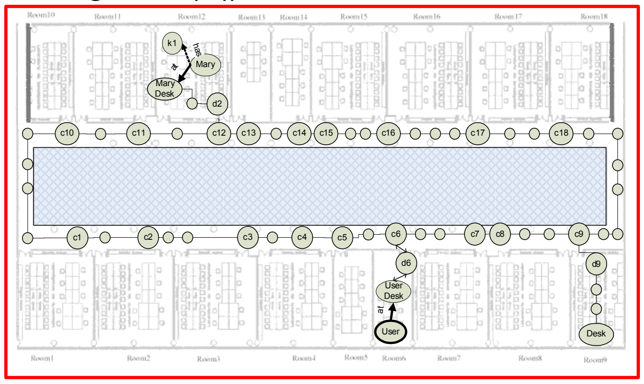

Figure 10. Scenario where the involved ground symbols in the final plan accepted by the user are shown. For clarity in the picture, symbols that represent the entrance of offices have been rewritten as "cX", being "X" the identifying office number, thus for instance, "c9" represents "entrance of office 9". Small circles represent intermediate distinctive places for navigation.

This solution is (at last!) accepted since it fulfills all the user's requirements and preferences. The "conversation" that has

[12] When communicating plans, sequences of navigational tasks can be abbreviated indicating the sequence of the destinations, as shown in this example.

taken place between the user and the robot has been realized using familiar concepts (symbols) of the former, which has produced an efficient interaction from the user's perspective. Also, the planning process has produced plans efficiently in spite of being confronted with a large and complex scenario.

V. Conclusions

In this paper we have proposed a symbolic model of the environment that makes human-machine interactive task planning affordable in the real world. The use of multiple abstraction permits a semi-autonomous agent to deal with large amounts of information and with task planning efficiently. It also provides each user with the best set of symbols (those that the user understands well and are according to his/her own experience) without compromising that efficiency. We have proposed a formalization of this multi-hierarchical model and of the task planning process. Based on that, we have also presented a form of interactive task planning that permits a user and a robot to collaborate at very different moments of the planning process in order to improve the solutions. We have illustrated this with simulated and real experiences.

Further work will be devoted to introducing non-classical planners into our approach (probabilistic methods) and using it for robots that are able to operate during long periods of time in crowded scenarios. We will also investigate automatic processes for learning the user preferences from his/her interaction with the system which could serve to regulate the amount of information reported to the user.

References

[1] Coradeshi S., and Saffiotti A., *An Introduction to the Anchoring Problem*. Robotics and Autonomous System vol. 43, No. 2-3, 2003, pp. 85-96.

[2] Harnand S., *The Symbol Grounding Problem*. Physica D: Nonlinear Phenomena vol. 42 (1990) pp. 335-346.

[3] Holding C.S., *Further Evidence for the Hierarchical Representation of Spatial Information*. Journal of Environmental Psychology, vol. 14, no.2, pp. 137-147, 1994.

[4] Hirtle, S.C. and Jonides J., *Evidence of Hierarchies in Cognitive Maps*. Memory and Cognition vol. 13, no.3,1985, pp. 208-217.

[5] Galindo C., Fernandez J.A., and Gonzalez J., *Control Architecture for Human-Robot Integration: Application to a Robotic Wheelchair*. IEEE Trans. on Syst, Man, and Cyb. (B), vol. 36, no.5, 2006, pp.1053-1067.

[6] Galindo C., Gonzalez J., and Fernandez J.A., *Hierarchical Task Planning through World Abstraction*. IEEE Trans. On Robotics vol. 20, no. 4, 2004, pp. 667-690.

[7] Fernandez J.A., and Gonzalez J., *Multi-Hierarchical Representation of Large-Scale Space* (Kluwer Academic Publishers, 2001.

[8] Fernandez J.A., Galindo C., and Gonzalez J., *Assistive Navigation of a Robotic Wheelchair using a Multihierarchical Model of the Environment*. Integrated Computer-Aided Engineering vol. 11, no. 11, 2004, pp. 309-322.

[9] Fernandez J.A., and Gonzalez J., *Multihierarchical Graph Search*. IEEE Trans. on Pattern Analysis and Machine Intelligence vol. 24, no. 1, 2002, pp. 103-113.

[10] Korf R.E., *Planning as Search: A Quantitative Approach*. Artificial Intelligence, vol. 33, 1987, pp. 65-88.

[11] Tversky B. *Cognitive Maps, Cognitive Collages, and Spatial Mental Models*. Lecture Notes in Computer Science, vol. 716, 1993, pp. 14-24.

[12] Sandberg E.H. *Constraints on the Development of Hierarchical Spatial Representations.* Cognitive Development, vol. 14, 1999, pp. 597-619.

[13] Voicu H. *Hierarchical Cognitive Maps*. Neural Networks, vol. 16, 2003, pp. 569-576.

[14] Blum A.L. and Furst M.L. *Past Planning through Planning Graph Analysis*. Artificial Intelligence, vol. 90, 1997, pp. 281-300.

[15] Jorg H. and Bernhard N. *The FF planning system: Fast Plan Generation through Heuristic Search*. J. Artificial Intelligence Research, vol. 14, 2001, pp. 253-302.

[16] Rydeheard D.E. and Burstall R.M. *Computational Category Theory*. Prentice Hall International (UK) Ltd., 1988.

[17] Patalano A.L., Smith E.E., Jonides J., and Koeppe R.A. *PET Evidence for Multiple Strategies of Categorization*. Cognitive, Affective, and Behavioral Neuroscience, vol. 1, no. 4, 2001, pp. 360-370.

[18] 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 and New York, Plenum Press, 1983.

[19] Kuipers B.J. *The Spatial Semantic Hierarchy*. Artificial Intelligence, vol. 119, 2000, pp. 191-233.

[20] Trudeau M.R. *Introduction to Graph Theory*. Dover Publications, 1993.

[21] Kennaway R. *Graph Rewriting in some Categories of Partial Morphisms.* Proc. 4th Int. Workshop on Graph Grammars and their Applications to Computer Science, vol. 532.Hartmut Ehrig, Hans-Jorg Kreowski and Grzegorz Rozengerg Eds., 1991, pp. 490-504.

[22] Sacerdoti E.D. *Planning in a Hierarchy of Abstraction Spaces*. Artificial Intelligence, vol. 5, no. 2, 1974, pp. 115-135.

[23] Knoblock C.A. *Automatically Generating Abstractions for Planning*. Artificial Intelligence, vol. 68, no. 2, 1994, pp. 243-302.

[24] Pineau J, Roy N, and Thrun S. *A Hierarchical approach to POMDP Planning and Execution*. Workshop on Hierarchy and Memory in Reinforcemente Learning (ICML), 2001.

[25] C. Galindo, J.-A. Fernández-Madrigal, J. González, A. Saffiotti, and P. Buschka, Life-Long Optimization of the Symbolic Model of the Environment of a Mobile Robot**,** *IEEE Transactions on Systems, Man, and Cybernetics part B (to appear).*

[26] Gonzalez J., Muñoz A.J., Galindo C., Fernandez-Madrigal J.A., and Blanco J.L. *A Description of the SENA Robotic Wheelchair*. Proc. 13th IEEE Mediterranean Electrotechnical Conf., May 2006, Spain.

[27] Giunchiglia F. and Walsh T. *A Theory of Abstraction*. Artificial Intelligence, vol. 57, no. 2-3, 1992, pp. 323-389.

[28] Ghorbel M., Haariz M., Grandjean B., and Mokhtari M. *Toward a generic human machine interface for assistive robots: the AMOR project*. 9th Int. Conf. on Rehab. Rob., 2005 ICORR 2005, pp. 168-172.

[29] Walsdorf A. and Onken R. *Cognitive man-machine-cooperation: modelling operators' general objectives and its role within a cockpit assistant system.* IEEE Int. Conf on Systems, Man, and Cyb., 2000 pp. 906-913.

[30] Murphy, R.R., Lisetti, C.L., Tardif, R., Irish, L., and Gage, A. *Emotion-based control of cooperating heterogeneous mobile robots*. IEEE. Trans. on Robotics and Automation, vol. 18, no. 5, 2002, pp.744-757.

[31] Barea R., Boquete L., Bergasa L.M., Lopez E., and Mazo, M. *Electro-Oculographic Guidance of a Wheelchair Using Eye Movements Codification*. Int. J. of Robotics Resarch, vol. 22, no. 7-8, 2003, pp. 651-652.

[32] Fernandez J.L., Simmons R., Sanz R., and Dieguez A.R. *A Robust stochastic supervision architecture for an indoor mobile robot.* Proc. of Int. Conf. on Field and Service Robotics, 2001.

[33] Fong T.W. and Thorpe C. *Vehicle teleoperation interfaces*. Autonomous Robots, vol. 11, no.1 pp. 9-18, 2001.

[34] Bourhis G., Horn O., Habert O., and Prusky A. *An autonomous vehicle for people with motor disabilities*. Robotics and Automation Magazine, vol. 8, no.1, 2001.

[35] Scholtz J. *Theory and evaluation of human-robot interaction*. 36th International Conference on System Sciences, Hawai, 2003.

[36] Nishikawa A., Hosoi T., Koara K., Negoro D., Hikita A., Asano S., Kakutani H., Miyazaki F., Sekimoto M., Yasui M., Miyake Y., Takiguchi S., and Monden M. *Face MOUSe: A novel human-machine interface for controlling the position of a laparoscope*. IEEE Trans. on Robotics and Automation, vol. 19, no.5, pp. 825-841, 2003.

[37] Borisoff J.F., Mason S.G., and Birch G.E. *Brain interface research for asynchronous control applications*. IEEE Trans. On Neural Systems and Rehab. Eng. *Vol. 14, no.2, pp. 160-164, 2006.*

[38] Moore M.M. *Real-world applications fro brain-computer interface technology*. IEEE Trans. On Neural Systems and Rehab. Eng. *Vol. 11, no.2, pp. 162-165, 2003.*

[39] Stiehl W.D., and Breazeal C.. *Affective Interaction and systems and applications- affective touch for robotic companions*. LNCS 2005, vol. 3784, pp. 747-754.

[40] Nudehi S.S., Mukherjee R., and Ghodoussi M. *A shared-control approach to haptic interface design for minimally invasive telesurgical training*. IEEE. Trans. on Control Systems Technology, vol. 13, no.4, pp. 588-592, 2005.

[41] Galindo C., Saffiotti A., Coradeschi S., Buschka P., Fernandez-Madrigal J.A., and Gonzalez J. *Multi-Hierarchical Semantic Maps for Mobile Robotics*. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Alberta (Canada), pp. 3492-3497, 2005

[42] Blanco J.L., Fernandez-Madrigal J.A., and Gonzalez J. *An entroypy-based measurement of certainty in Rao-Backwellized particle filter mapping*. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Beijing (China), Sept 9-15, 2006.

[43] Agah A. and Tanie K. *Human Interaction with a Service Robot: Mobile-Manipulator handing over and object to a human*. IEEE ICRA, pp. 575-580, New Mexico, 1997

[44] Kulic D. and Croft A. *Strategies for safety in human robot interaction*. Proc.11th Int. Conf. on Advanced Robotics 2003, pp. 644-649, Coimbra.

[45] KangWoo L, Hyoung K. Wan C.Y., Young Y, and Dong K. *Designing a human-robot interaction framework for home service robot*. IEEE Workshop on Robots and Human Interactive Communication, 2005.

[46] Thrun S. *Toward a framework for human-robot interaction*. Human-Computer Interaction, vol. 19, pp. 9-24, 2004.

[47] Hyoung K., KanWoo L, and Dong K. *Emotional interaction model for a service robot*. IEEE Workshop on Robots and Human Interactive Communication, pp. 672,678, 2005.

[48] Marrone F., Raimondi F., and Strobel M. *Compliant interaction of a domestic service robot with a human and the environment*. Proc. Of 33rd Int. Symposium on Robotics, Stockholm, 2002.

[49] Hwang Y.K., Chen P.C., and Watterberg P.A. *Interactive Task Planning through Natural Language.* ICRA, Minneapolis (Minnesota), 1996.

[50] Dillmann R., Zoellner R., Ehrenmann M., and Rogalla O. *Interactive Natural Programming of Robots: Introductory Overview.* Proc. of DREH, Tolouse (France), 2002.

[51] Sun H., Hujun B., Man T.N., and Fai W.L. *Interactive Task Planning in Virtual Assembly.* Proceedings of the ACM symposium on Virtual reality software and technology, London (UK), 1999.

[52] Bagchi S., Biswas G., and Kawamura K. *Interactive task planning under uncertainty and goal changes* . Robotics and Autonomous Systems, vol. 18, pp. 157-167, 1996.

[53] Holding C.S. *Further Evidence for the Hierarchical Representation of Spatial Information*. J. of Environmental Psychology, vol. 14, n. 2, 1994, pp. 137-147.

[54] Hirtle S.C. and Jonides J. *Evidence of Hierarchies in Cognitive Maps*. Memory and Cognition, vol. 13, n. 3, 1985, pp. 208-217.

[55] McNamara T.P., Hardy J.K. and Hirtle S.C. *Subjective Hierarchies in Spatial Memory*. J. of Experimental Psychology: Learning, Memory, and Cognition, vol. 15, 1989, pp. 211-227.