# Belief Management for HRI Planning

Julien Guitton, Matthieu Warnier, Rachid Alami

# Belief Management for HRI Planning

**Julien Guitton** and **Matthieu Warnier** and **Rachid Alami** [1]

**Abstract.** This paper presents an extension of a hierarchical planning approach designed to handle multi-agent problems and, more especially, Human-Robot Interaction problems in which a robot and a human have to collaborate in order to achieve a joint goal. Our method allows to reason and plan for agents that have different or incomplete beliefs in order to produce feasible and comprehensible plans. It is based on a new description of the agent's beliefs and a mechanism that produces and inserts some communication actions into the current plan.

## 1 Introduction

When acting in an environment with other partners, an agent has to reason not only on its own capabilities but also on the capabilities of the other agents to achieve a task in a collaborative way. In the context of Human-Robot Interaction (HRI), the robot needs to reason about the human's knowledge: the robot and the human may not have the same vision of the scene as well as the same information about the objects of the environment. This reasoning is complicated by the fact that the robot may not have the cognitive model of the human. Indeed, except through some dialog phases, the robot can only infer the knowledge of its human partner concerning the environment through a reasoning using a perspective-taking ability.

With this knowledge, which can be different from its own knowledge or partially incomplete, the robot has to produce a plan for him and for its partner in order to achieve a joint goal. This plan should be precise and comprehensible for the human.

In a previous work, we have presented a dedicated planner called HATP [2], for Human Aware Task Planner, which is based on hierarchical task planning combined with a set of behavior rules that leads the robot decisions and allows to produce plans that are socially acceptable for humans. In this paper, we extend this planner to deal with HRI problems in which the robot and the human may not have the same beliefs on the environment or incomplete beliefs. We call this extension *Belief Management*.

In the next section, we make an overview of existing work on the consideration of the human when designing a robotic architecture and existing work on planning for collaborative task achievement between a robot and a human. Then, in section 3, we present the HATP planner which provides the basis for this work. In section 4, we propose an extension of the HATP formalism to take into account beliefs of the different partners and the algorithm part allowing to handle this extension. In section 5, we present the integration of this work in our robotic platform as well as the different modules allowing to gather and manage agent's knowledge. Finally, in section 6 and 7, we illustrate the planning process with Belief Management through some basic examples and a scenario in real situation where a human and a robot have to cooperate in order to clean a table, *i.e.,* to put some tapes into a trash bin.

[1] CNRS, LAAS, 7 avenue du colonel Roche, F-31400 Toulouse, France; Univ. de Toulouse, LAAS, F-31400 Toulouse, France; email: firstname.lastname@laas.fr

## 2 Context and related work

In recent years, the Human Robot Interaction field has become an active research topic in various disciplines and at different levels. For instance, for researchers in sociology, one of the current trends is to evaluate the reactions of humans interacting with robots [10] in order to design more friendly-user robotic architectures.

In robotics, the human is taken into account at different levels such as at the perception layer through some work on perspective taking [9, 18, 19] or at the functional layer in order to adopt a socially acceptable behavior during motions by considering the human not only as an obstacle to avoid [16].

Another trend in robotics and HRI field is to develop cognitive architectures that try to be as close as possible to the cognitive model of humans [6, 8, 11]. The idea behind these cognitive architectures is to embed in the robot architecture a theory of mind [3], *i.e.* the ability for the robot to infer and understand the beliefs, desires and intentions of others from its observations.

At the decision layer, work on planning for HRI has follow two main trends. The first approach concerns work on mixed-initiative planning [5, 14] that allows to put the human in the loop: the human can control the construction of a plan while the planner is used to assist him in making decisions. The other approach is called continual planning [4] and is based on the idea of active knowledge gathering [12]: the robot does not plan only to achieve a goal, but also to acquire the necessary information to achieve it. Continual planning interleaves planning and execution in order to compensate the lack of information from a planning phase to another.

In this work, we consider the human only at the deliberative level, and more especially at the planning level. Unlike continual planning, in order to avoid re-planning and produce comprehensible plans, our planning algorithm reasons from not only the robot's knowledge about the environment but also from its knowledge concerning the human's beliefs. When the lack of information concerns the robot's beliefs, the algorithm behaves like continual planning by acquiring the information and trying to solve the goal again.

## 3 Human Aware Task Planner

HATP, for Human-Aware Task Planner, is a HTN planner. The aim of hierarchical task planning is to decompose a high-level task representing a goal into sub-tasks until reaching atomic tasks that are achievable by the agents [15]. HATP is able to produce plans for the robot's actions as well as for the other participants (humans or robots). It can be tuned by setting up different costs depending on

the actions to apply and by taking into account a set of constraints called social rules. This tuning aims at adapting the robot's behavior according to the human's preferences and to the desired level of cooperation.

### 3.1 Agents and action streams

The robot plans not only for itself but also for the other agents. The resulting plan, called "shared plan" is a set of actions that forms a stream for each agent involved in the goal achievement. Depending on the context, some shared plans contain causal relations between agents. For example, the second agent needs to wait for the success of the first agent's action to be able to start its own action. When the plan is performed, causal links induce some synchronizations between agents. Figure 1 illustrates a plan with two streams.

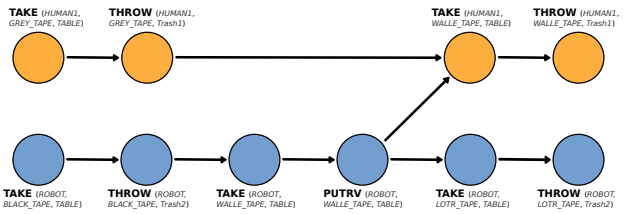



**Figure 1.** A plan produced by HATP with 2 streams.

### 3.2 Action costs and social rules

To each action is associated a cost function and a duration function. The duration function provides a duration interval for the action achievement and is used, on the one hand, as a timeline to schedule the different streams and, on the other hand, as an additional cost function. In addition to these costs, HATP takes as an entry a set of social rules. Social rules are constraints aiming at leading the plan construction towards the best plan according to some human's preferences. The main social rules we have defined are:

- undesirable state. To avoid a state of the world in which the human could feel uncomfortable;
- undesirable sequence. To eliminate sequences of actions that can be misinterpreted or rejected by the human;
- effort balancing. To adjust the work effort between agents;
- wasted time. To avoid delays between the actions of an agent;
- intricate links. To limit dependencies between the actions of two or more agents.

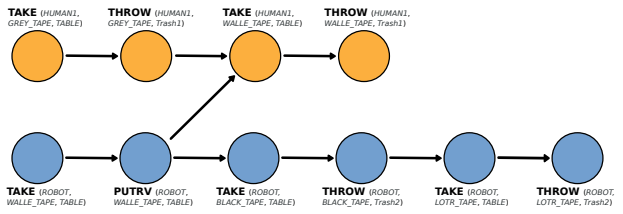



**Figure 2.** A plan with the wasted time social rule.

Figure 2 illustrates an alternative plan to the previous one (Figure 1) if the social rule called wasted time is used. The returned plan is the best plan according to a global evaluation of these multiple criteria. In this plan, we can see that the actions of the robot are reordered in order to remove the waiting period of the human.

### 3.3 Several levels of cooperation

By tuning its costs and applying social rules, HATP can be used to compute various alternative plans. These plans can be categorized into several levels of cooperation:

- helping the human to achieve his goal by acting for him;
- sharing concrete resources by handing some objects;
- collaboration of the robot and the human by coordinating their actions towards a human-robot joint goal.

### 3.4 Domain modeling

HATP uses its own object-oriented domain modeling language. This language has at least the same expressive power and features than SHOP2 [15]. In order to better understand the belief management extension, we present the basis of this modeling language.

The world is represented by a set of entities. Each entity is unique and is defined by a set of attributes. Attributes are defined to be *static* or *dynamic* and have a type *atom* or *set*. A *static* attribute represents a non-modifiable information whereas a *dynamic* attribute can be updated. An attribute of type *atom* can take only one value at a time whereas the *set* type is used to store multiple values.

An agent is a special object and therefore is defined using the same formalism. However, as the output of HATP is a plan under the form of a stream per agent, the agent entity type is predefined and at least one agent must be initialized.

The domain, called *fact database* in HATP, is processed in four steps as illustrated in figure 3. First, the different types of entities are defined (except for Agent which is implicit). Then, the attributes of each entity are defined. In a third step, the objects and agents present in the environment are created. Finally, initial values are given to the attributes of each entity.

## 4 Belief management and formalism

The description of the world as it has been previously presented assumes that the current state is entirely known and that all the agents share the same view of this world. In a real application, especially in Human-Robot Interaction problems, these assertions can lead to unfeasible solutions or illogical or impracticable plans for the human.

In order to bridge this gap, we propose to extend this representation by adding, on the one hand, the possibility to model a different knowledge for each agent and, on the other hand, the possibility to consider that an agent may or may not know some information.

### 4.1 Belief state modeling

In our HRI experiments, the solution to a given goal is entirely computed by the robot. In this case, the *fact database* must model the state of the world from the robot's point of view as well as the robot's beliefs concerning the human's (or more generally the other agents) knowledge.

```
factdatabase {
  //step 1: Definition of entity types
  define entityType Table;
  define entityType Container;
  define entityType GameArtifact;
  //step 2: Definition of attributes
  define entityAttributes Agent {
    static atom string type;
    dynamic atom GameArtifact hasInRightHand;
  }
  define entityAttributes Container {
    dynamic set Agent isReachableBy;
  }
  define entityAttributes GameArtifact {
    dynamic set Agent isVisibleBy;
    dynamic set Agent isReachableBy;
    dynamic atom Container isIn;
    dynamic atom Table isOn;
  }
  //step 3: Creation of entities
  JIDO = new Agent;
  PINK_TRASHBIN = new Container;
  WHITE_TAPE = new GameArtifact;
  //step 4: Attributes initialization
  JIDO.type = "robot";
  PINK_TRASHBIN.isReachableBy <<= JIDO;
  WHITE_TAPE.isVisibleBy <<= JIDO;
  WHITE_TAPE.isReachableBy <<= JIDO;
  WHITE_TAPE.isIn = PINK_TRASHBIN;
}
```

**Figure 3.** Example of domain definition using the HATP formalism.

#### 4.1.1 Belief representations:

To model different beliefs for the agents, the HATP formalism is extended using Multiple Values State Variables (MVSV). A multiple values state variable $V$ is instantiated from a domain $Dom$ and for each agent $a \in A$ the variable V has an instance $V(a) = v \in Dom$. For example, if the agents have a different belief about the location of WHITE_TAPE:

```
WHITE_TAPE(JIDO).isIn = PINK_TRASHBIN;
WHITE_TAPE(HERAKLES).isIn = BLUE_TRASHBIN;
```

By default, in order to clarify the planning domain, only entity attributes for which the agents have a different belief are modeled with the MVSV formalism.

In order to specify which agent is the robot, *i.e.,* the agent for which the system plans, we use the keyword **myself** instead of declaring a new agent. For example, if JIDO is the robot and HERAKLES is a human, the initialization will look like:

```
JIDO = myself;
HERAKLES = new Agent;
```

#### 4.1.2 known and unknown information:

The agent's beliefs model includes the notion of *known* and *unknown* information. When an agent has no information about an entity attribute, the value of this property is set to *unknown*. When an agent, different of the agent **myself** knows the value of an attribute, this value is set to *known*.

The previous belief representation is extended to take into account these specific values:

$$V(a) \in \begin{cases} Dom_v \sqcup \{unknown\} & \text{if } a = \text{myself} \\ Dom_v \sqcup \{unknown\} \sqcup \{known\} & \text{otherwise} \end{cases}$$

For example, if the human doesn't know the location of the object WHITE_TAPE:

```
WHITE_TAPE(JIDO).isIn = PINK_TRASHBIN;
WHITE_TAPE(HERAKLES).isIn = unknown;
```

When an agent has no information about an entity, *i.e.,* all the attributes of this entity should be set to unknown, we simplify the representation by:

```
WHITE_TAPE(HERAKLES) = unknown;
```

With this representation, we assume that even if the agent has no information on the object, it knows the existence of the object.

#### 4.1.3 Consistency of beliefs:

To be consistent, a belief on a state variable $V$ requires that the union of the agent's beliefs forms a set of dimension 1. That is to say, a value is consistent if there is no conflict between the agents' beliefs concerning the object.

$$\| \bigcup_{\forall a \in Ag} V(a) \| = 1$$

This property is used during the planning process in order to assume a correct plan in the point of view of the agent's beliefs.

### 4.2 Beliefs update and communication

In classical planning, an action is defined by a set of preconditions representing the necessary conditions to achieve it and a set of effects modeling the resulting changes of the world.

Planning for several agents that have their own beliefs raises some fundamental questions:

- Which beliefs should trust the system, especially in the case of a joint action?
- Should the agent's beliefs be consistent before an action achievement? And how?
- How evolve beliefs of the agents involved in a joint action?
- How evolve beliefs of the other agents?

In the following paragraphs, we try to answer these questions by specifying the behavior of a classical action and by introducing a specific type of actions called *communication actions*.

#### 4.2.1 action preconditions and effects:

In order to achieve a joint action, every participants must have the same beliefs about the entities manipulated during this action. To ensure this consistence, the planner will produce some communication actions between the main agent (myself) and the other participants.

Because when beginning to achieve an action, all the participants may have the same beliefs on the manipulated objects, the effects of the action are applied over the beliefs of all participant, *i.e.,* beliefs on the manipulated objects remain consistent after the action achievement. Concerning the other agents that could be present in the scene during this action achievement, it is the responsibility of the domain designer to decide if their beliefs should be updated. For example, such an update could be:

```
FORALL(Agent O, {O != A;}, {C(O).isIn = C(myself).isIn;})
```

Meaning that for all agents O distinct from the agent A doing the action, their value of the property `isIn` for the object C is updated with the value of the property `isIn` of C stored in the agent myself database. Beliefs of the agent A are updated automatically according to the classical effects of the action.

#### 4.2.2 Communication actions:

A communication action is a specific action that takes as parameters two agents, the emitter and the receiver, and a subject which is represented by an entity and an attribute.The prototype for a communication action is:

```
commAction name(Agent A, Agent B, Entity E, Attribute T){
  preconditions {...};
  effects {...};
  cost{...};
  duration{...};
}
```

The aim of a communication action is to transmit a value from one agent to another and corresponds to the effect :

```
E(B).T = E(A).T;
```

This effect is implicit to this kind of action, *i.e.*, the domain designer does not need to specify it.

Like a classical action, a communication action is defined by a set of preconditions to express the necessary conditions to achieve the communication (*e.g.,* the agents must be in the same room), and a set of additional effects. With these effects, it is possible to model the concept of co-presence, *i.e.*, the communication affects also the beliefs of all the agents that are listening.

In order to fit the domain formalism, if the parameter corresponding to the attribute is set to `NULL`, then all the attributes of the entity E are transmitted from the agent A to the agent B. Otherwise, only the value of the specified attribute is communicated.

#### 4.2.3 Types of communication:

Depending on the agents' beliefs, the communication acts will not be treated the same way at the execution level. We choose to make this distinction at the planning level by defining three different communication actions: *information*, *contradiction* and *question*. This distinction may help, during the plan execution, to choose the best communication modality to apply.

The communication action of type *information* aims at giving an information from the agent myself to another agent when the value for an attribute is set to *unknown* in its knowledge base.

When the value associated to an attribute for an agent is different of the value for the agent myself, the planner produces a communication action of type *contradiction*.

The *question* type is used when the agent myself has no information concerning an attribute and another agent has this information (modeled by the value *known*).

| myself | $\text{Agent}_B$ | type of communication |
|---|---|---|
| $v$ | $unknown$ | information |
| $v$ | $v'$ | contradiction |
| $unknown$ | $known$ | question |

**Table 1.** Types of communication depending on the agents' beliefs.

Table 1 summarizes these types of communication depending on an agent's beliefs compared to the beliefs of the agent myself.

### 4.3 Implementation and adaptation of the planner

To be able to deal with the agents' beliefs, the planning algorithm must be adapted to take into account the new formalism and the communication actions.

#### 4.3.1 Fact databases:

To store the agents' beliefs, we create a fact database for each agent. During the initialization phase, the entities representing the agents and objects in the environment are created and stored in the database of the agent myself. For the other agents, in order to save memory, we decide to store in the additional agents' databases only the values that are inconsistent with the beliefs of the agent myself.

#### 4.3.2 General communication method:

In order to let the planning domain designer name the communication actions as he would do for classical actions, the communication actions are linked to the concepts of *information*, *contradiction* and *question* through a method called **beliefManagement**.

```
beliefManagement {
  information { GiveInformationAbout; };
  contradiction { ForceInformation; };
  question { AskForInformation; };
}
```

Each communication action must have been defined previously in the planning domain.

During planning, when a communication is needed, this method returns the appropriate communication action depending on the needed communication type.

#### 4.3.3 Main planning process:

The main algorithm of the HTN planner consists in developing the planning tree, *i.e.,* in decomposing complex tasks into sub-tasks until reaching a sequence of primitive actions. The tree development is done by a depth-first search and stops when the task list is empty.

This algorithm is enhanced with the belief management processing as follow:

```
1  Tree_develop(T):
2  t0 = T[0];
3  if(t0 is a primitive task) {
4    classical = false;
5    agent = the (main) agent achieving the action;
6    v(agent)= value of task arguments and preconditions;
7    forall(v(agent)) {
8      if(agent ≠ myself) {
9        if(v(agent) = unknown) {
10         c = make_action(beliefManagement.information);
11         T[0] = c;
12         Tree_develop(T);
13       }
14       else if(v(agent) ≠ v(myself)) {
15         c = make_action(beliefManagement.contradiction);
16         T[0] = c;
17         Tree_develop(T);
18       }
19       else classical = true;
20     else if(v(myself)=unknown and v(other agent)=known) {
21       c = make_action(beliefManagement.question);
22       T = empty;
23       plan = c;
24     }
25     else classical = true;
26   }
27   if(classical=true) {
28     // do the classical HTN treatment for primitive task
29   }
30 }
31 else // do the classical HTN treatment for compound task
32 }
```

In this algorithm, if the current task corresponds to an action (l.3), the values of the attributes of each object linked to the task are verified. If the agent achieving the action is not myself (l.8) and if the value of an attribute is *unknown* (l.9) in the knowledge of this agent, then a communication action of type *information* is produced (l.10).

This action is inserted at the beginning of the task list (l.11) and will be refined during the next recursive call of the algorithm (l.12).

In the same way, if the value of the attribute in the model of the agent achieving the action is different from the value in the model of myself (l.14), a communication action of type *contradiction* is produced (l.15) and inserted at the beginning of the task list.

If the agent achieving the action is the myself agent and if the value of the attribute is *unknown* (l.20), the algorithm verifies that another agent knows this data. If it is the case, the algorithm produces a communication action of type *question* (l.21) and replaces all the pending tasks by this action (l.22 and l.23). Indeed, only after the execution of this communication action, the knowledge model of the main agent will be updated, then the planner would produce the remaining actions allowing to achieve the current goal during the re-planning phase.

# 5 Integration in a robotic architecture

HATP with Belief Management has been integrated and tested in our robotic architecture. In this section, we make a brief overview of this architecture.

## 5.1 Overview of the robotic architecture

The robot is controlled by a three-layer architecture [1]. Figure 4 illustrates the decisional layer of this architecture. The proposed decisional framework consists of several modules, having each a specific role and that can be linked to the three main activities of the robot controller: 1. Situation assessment and context management, 2. Goals and plans management, 3. Action refinement, execution and monitoring.

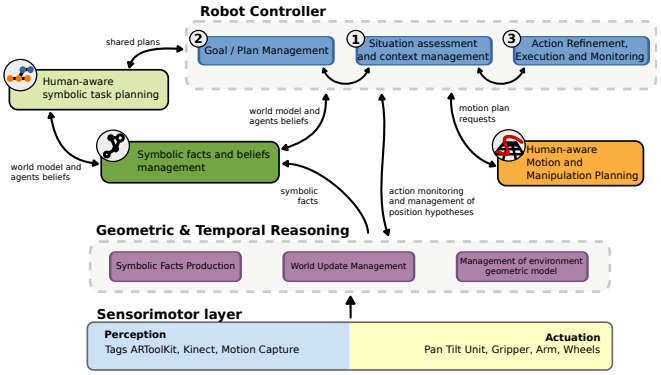



**Figure 4.** The decisional layer of the robotic architecture.

In the next paragraphs, we present parts of these activities that allow HATP to gather necessary knowledge in order to be able to produce a plan, and how this plan is executed.

## 5.2 Knowledge acquisition

The geometric reasoning component is called SPARK (SPAtial Reasoning and Knowledge) [18]. It is responsible for geometric information gathering and embeds a number of decisional activities linked to abstraction and inference based on geometric and temporal reasoning. SPARK maintains all geometric positions and configurations of agents, objects and furniture coming from perception and previous or *a priori* knowledge. Geometric states of the world are abstracted into a set of symbolic facts that can be directly used by HATP.

These produced facts are stored in a central symbolic knowledge base, called ORO [13]. ORO stores independent knowledge models for each agent. The robot architecture components can then save the different beliefs in the corresponding model. Each of these models is independent and logically consistent, enabling reasoning on different perspectives of the world that would otherwise be considered as globally inconsistent (for instance, a object can be visible by an agent but not by the others).

## 5.3 Goal treatment, planning and execution

The goal is given by the human partner. When an event announcing the goal is caught by the robot controller, its validity is first tested: does it correspond to abilities of the agents? Is it not already achieved?

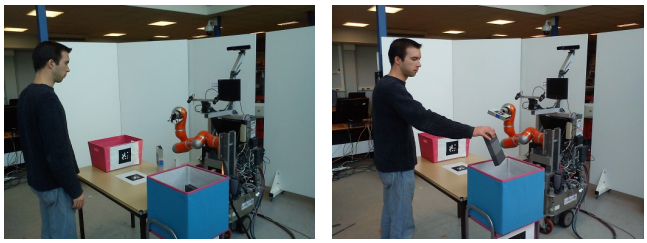
Then the goal is sent to HATP which acquires the current state of the world for each agent from ORO and produces a first plan. This goal is considered as achievable as long as the planner computes a valid plan or the execution is not abandoned by the human.

Plan execution consists in the management of all the actions of the plan. This management is done in three steps: first, the action preconditions are tested. Then, the action is executed and monitored (only monitored for human's actions). Finally, the expected effects are verified in order to acknowledge the action achievement. In case of failure, a new plan is requested and executed.

# 6 Two illustrative examples

To illustrate HATP with Belief Management, we details two examples of the achievement of a collaborative task in which the knowledge is incomplete.

Two agents, a robot (called JIDO_ROBOT) and a human (HERAKLES_HUMAN) have to collaborate in order to clean a table (EXP_TABLE). The goal is to put two tapes (BLACK_TAPE and GREY_TAPE) into the pink trash bin. The grey tape is on the table whereas the black tape is in the blue trash bin (Figure 5).



(a) Initial state of the experiment (b) During the execution of the plan

**Figure 5.** Representation of the environment for the HRI experiments.

In both experiments, the pink trash bin is reachable by both agents, the blue trash bin is reachable only by the human and the grey tape is reachable only by the robot. The reachability of the blue trash bin induces the reachability of the black tape. All these facts are computed by the SPARK module. The initial state is defined as follow (except for the black tape):

```
BLUE_TRASHBIN.isReachableBy <<= HERAKLES_HUMAN;
PINK_TRASHBIN.isReachableBy <<= HERAKLES_HUMAN;
PINK_TRASHBIN.isReachableBy <<= JIDO_ROBOT;
GREY_TAPE.isVisibleBy <<= JIDO_ROBOT;
GREY_TAPE.isVisibleBy <<= HERAKLES_HUMAN;
GREY_TAPE.isReachableBy <<= JIDO_ROBOT;
GREY_TAPE.isOn = EXP_TABLE;
```

### 6.1 First example: Unknown for the human

In this first scenario, the environment has been set up during the absence of Herakles. We assume that, from its standing position, the human cannot see the content of the blue trash bin, resulting in the fact that the location of the black tape is unknown for him. The initial state is extended as follow:

```
BLACK_TAPE(JIDO_ROBOT).isIn = BLUE_TRASHBIN;
BLACK_TAPE(HERAKLES_HUMAN).isIn = unknown;
```

Figure 6 illustrates the plan produced by HATP for this scenario. The first action is a communication action of type *information*. The robot informs the human that the black tape is in the blue trash bin. Then, the human has to pick and throw this tape into the pink trash bin while the robot has to pick and throw the grey tape.

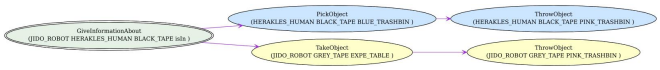



**Figure 6.** Plan produced for the first scenario. The robot informs the human about the location of the grey tape.

### 6.2 Second example: Unknown for the robot

In this second experiment, the perception system of the robot has been deactivated during the placement of the objects. For Jido, The location of black tape is unknown. It only knows that Herakles knows where is the tape. The initial state is extended as follow:

```
BLACK_TAPE(JIDO_ROBOT).isIn = unknown;
BLACK_TAPE(HERAKLES_HUMAN).isIn = known;
```

Because the robot only knows that the human knows where is the black tape, it cannot produce a complete plan for the given goal. Indeed, this information is needed by the preconditions of the action PickObject. In this case, HATP produces a plan containing only one action (figure 7): a communication action of type *question* allowing the robot to gather the missing knowledge.


**Figure 7.** The plan contains only a communication action allowing the robot to acquire the missing information.

Once the robot has the information, HATP is asked to compute a new plan. Figure 8 illustrates the ouput of HATP allowing the agents to achieve the goal.

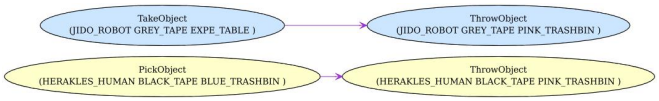



**Figure 8.** Plan produced after the acquisition of the information and a re-planning step.

## 7 A more complete scenario

In this scenario, the experimental conditions are slightly different from the previous examples. Both agents have to throw three tapes (BLACK_TAPE, GREY_TAPE and WHITE_TAPE) that are on the table, into the pink trash bin. The white tape is only accessible by the robot whereas the grey and black tapes are reachable only by the human but invisible to him because they are hidden behind some boxes (BOX1 and BOX2). Because the pink trash bin is only reachable by the human, he is in charge of throwing the three tapes.

Figure 9 is a screenshot of the SPARK module and illustrates the initial state of this scenario, that is to say the representation of the environment modeled from the robot's point of view.

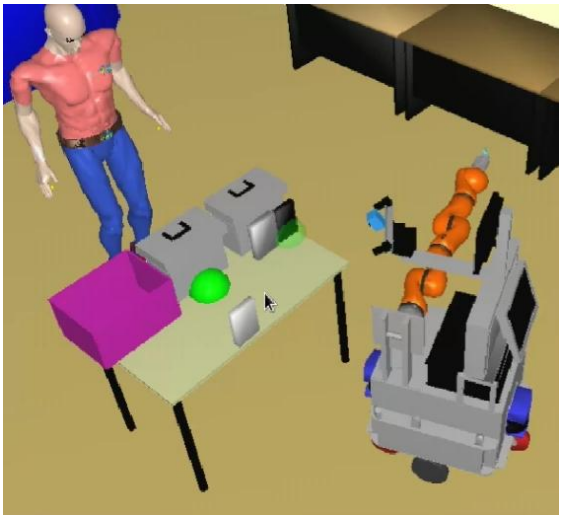


**Figure 9.** View of the initial state computed by the SPARK module.

The human agent, Herakles, has no information on the position of the black tape and believes that the grey tape is behind the central box (BOX1). This belief is represented by the green sphere. In fact, this grey tape is positioned behind the other box (BOX2), as represented on the figure 9.

The plan produced by HATP corresponds to the one on the figure 10. This plan contains two communication actions (figure 11): an information on the existence of the black tape and a contradiction on the position of the grey tape.

One can remark that the attribute communicated by the robot for the action of type *information* is set to `NULL` resulting in the fact that all the attributes of the black tape are communicated to Herakles.

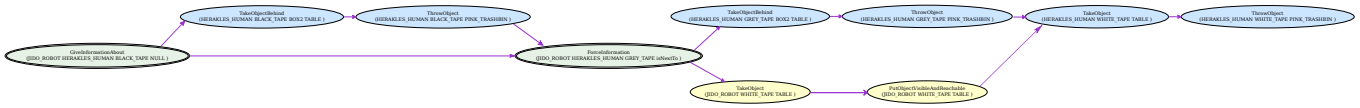


**Figure 10.** Plan produced by HATP for this cooperative scenario. This plan contains 2 communication actions. Actions of Herakles are in blue and actions of Jido are in yellow. Green circles stand for the communication actions.

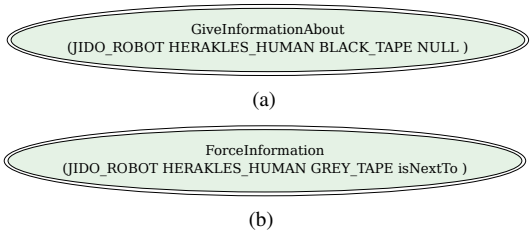



**Figure 11.** Zoom on the two communication actions.

## 8 Conclusion and future work

In this paper, we have presented a first attempt allowing a hierarchical task planner to produce valid and comprehensible plans for a human even if the agents have different beliefs or incomplete knowledge. This work is based on the extension of the HATP formalism allowing to express different or known/unknown beliefs for each agent, and the design of special actions: the communication actions. During the planning process, when the agents' beliefs are inconsistent or when one of the agents has not the necessary knowledge to achieve the action, a communication action of type contradiction, information or question is produced and inserted in the current plan.

This work has been implemented in the decisional architecture of our personal assistant robot and tested through some simple but realistic scenarios.

Our future work will concern the production of *known* and *unknown* facts, which has not been roughly implemented in our situation assessment module (SPARK). This work need some extra temporal and geometric reasoning about the human. Did he see or not the environment changing? what did he know before leaving?

The communication actions are, for now, only executed under the form of spoken sentences. We would like to investigate the possibility of using other modalities (gaze, gesture, ...) and to combine them.

Moreover, humans have a tendency to forget or not accept what they were told as truth. It would be interesting to see how this influences planning and execution of the plan. A justification of a modification of the environment given by the robot can also lead to a better acceptance for the human.

Concerning the planning part, one possibility to extend HATP in order to avoid re-planning even if the robot has missing information would be to apply techniques from planning under incomplete knowledge as for example in the work of Petrick and Bacchus [17], or following some work on acting in noisy environment [7].

## 9 Acknowledgment

This work has been conducted within the EU SAPHARI (Safe and Autonomous Physical Human-Aware Robot Interaction) project (http://www.saphari.eu/) funded by the E.C. Division FP7-IST under Contract ICT-287513.

## REFERENCES

[1] R. Alami, R. Chatila, M. Ghallab, and F. Ingrand, 'An architecture for autonomy', *Int. Journal of Robotics Reasearch*, **17**(4), 315–337, (1998).
[2] S. Alili, V. Montreuil, and R. Alami, 'HATP: Task planner for social behavior control in autonomous robotic systems for HRI', in *9th Int. Symposium on Distributed Autonomous Robotic Systems*, (2008).
[3] S. Baron-Cohen, 'Precursors to a theory of mind: understanding attention in others', *Natural theories of mind: Evolution, development and simulation of everyday mindreading*, **1**, 233–251, (1991).
[4] M. Brenner and B. Nebel, 'Continual planning and acting in dynamic multiagent environments', *Journal of Autonomous Agents and Multiagent Systems*, **19**, 297–331, (2009).
[5] J.L. Bresina, A.K. Jonsson, P.H. Moris, and K. Rajan, 'Mixed-initiative activity planning for mars rovers', in *19th international conference on Artificial Intelligence*, (2005).
[6] N. Cassimatis, *A cognitive architecture for integrating multiple representation and inference schemes*, Ph.D. dissertation, MIT, 2002.
[7] A. Gabaldon and G. Lakemeyer, 'ESP: A logic of only-knowing, noisy sensing and acting', in *AAAI*, pp. 974–979, (2007).
[8] L.M. Hiatt and J.G. Trafton, 'A cognitive model of theory of mind', in *Proceedings of the 10th International Conference on Cognitive Modeling*, pp. 91–96, (2010).
[9] L.M. Hiatt, J.G. Trafton, A.M. Harrison, and A.C. Schultz, 'A cognitive model for spatial perspective taking', in *the sixth International Conference on Cognitive Modeling*, pp. 354–355, (2004).
[10] A. Hiolle, K.A. Barde, and L. Canamero, 'Assessing human reactions to different robot attachment profiles', in *18th IEEE int. symposium on Robot and Human Interactive Communication*, pp. 251–256, (2009).
[11] D.E. Kerias and D.E. Mayer, 'An overview of the EPIC architecture for cognition and performance with application to human-computer interaction', *Human-Computer Interaction*, **12**, 391–438, (1997).
[12] C.A. Knoblock, 'Planning, executing, sensing and replanning for information gathering', in *14th international joint conference on artificial intelligence*, (1995).
[13] S. Lemaignan, R. Ros, L. Mosenlechner, R. Alami, and M. Beetz, 'ORO, a knowledge management module for cognitive architectures in robotics', in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, (2010).
[14] K.L. Myers, W.M. Tyson, M.J. Woverton, P.A. Jarvis, T.J. Lee, and M. Desjardins, 'PASSAT: A user-centric planning framework', in *Third international workshop on planning and scheduling for space*, (2002).
[15] D. Nau, T.-C. Au, O. Ilghami, U. Kuter, J. W. Murdoch, D. Wu, and F. Yaman, 'SHOP2: An HTN planning system', *Journal of Artificial Intelligence Research*, **20**, 380–404, (2003).
[16] A.K. Pandey and R. Alami, 'A framework for adapting social conventions in a mobile robot motion in a human-centered environment', in *International Conference on Advanced Robotics*, pp. 1–8, (2009).
[17] R.. Petrick and F. Bacchus, 'Extending the knowledge-based approach to planning with incomplete information and sensing', in *International Conference on Automated Planning and Scheduling*, (2004).
[18] E.A. Sisbot, R. Ros, and R. Alami, 'Situation assessment for human-robot interaction', in *20th IEEE International Symposium in Robot and Human Interactive Communication*, (2011).
[19] J.G. Trafton, N. Cassimatis, M. Bugajska, D. Brock, F. Mintz, and A. Schultz, 'Enabling effective human-robot interaction using perspective-taking in robots', *IEEE transactions on Systems, Man, and Cybernetics*, **35**(4), 460–471, (2005).