Human-Robot Interaction: Tackling the AI Challenges

Séverin Lemaignan, Mathieu Warnier, E. Akin Sisbot, Rachid Alami

CNRS, LAAS, 7 avenue du Colonel Roche, F-31400 Toulouse, France
Univ de Toulouse, LAAS, F-31400 Toulouse, France
firstname.lastname@laas.fr

Abstract

Human-Robot interaction is an area full of challenges for artificial intelligence: dynamic, partially unknown environments that are not originally designed for autonomous machines; a large variety of situations and objects to deal with, with possibly complex semantics; physical interactions with humans that requires fine, low-latency control, representation and management of several mental models, pertinent situation assessment skills...the list goes on.

This article sheds light on some key decisional issues that are to be tackled for a cognitive robot to share space and tasks with a human, and present our take on these challenges. We adopt a constructive approach based on the identification and the effective implementation of individual and collaborative skills. These cognitive abilities cover geometric reasoning and situation assessment mainly based on perspective-taking and affordances, management and exploitation of each agent (human and robot) knowledge in separate cognitive models, natural multi-modal communication, "human-aware" task planning, and human and robot interleaved plan achievement.

We present our design choices, the articulations between the diverse deliberative components of the robot, experimental results, and eventually discuss the strengths and weaknesses of our approach. It appears that *explicit* knowledge management, both symbolic and geometric, proves to be key as it pushes for a different, more *semantic* way to address the decision-making issue in human-robot interactions.

Keywords: human-robot interaction, cognitive robotics, perspective taking, knowledge representation and reasoning

1. The Challenge of Human-Robot Interaction

1.1. The Human-Robot Interaction Context

Human-robot interaction is a challenge for artificial intelligence. This field lays at the crossroad of several domains of AI and requires to tackle them in a holistic manner: modelling humans and human cognition; acquiring, representing, manipulating in a tractable way abstract knowledge at the human level; reasoning on this knowledge to make decisions; and eventually instantiating those decisions into physical actions both legible to and in coordination with humans. Many AI techniques are invited, from visual processing to symbolic reasoning, from task planning to *theory of mind* building, from reactive control to action recognition and learning.

We do not claim to address here the issue as a whole. This article attempts however to organise it into a coherent challenge for artificial intelligence, and to explain and illustrate some of the paths that we have investigated on our robots, that result in a set of deliberative, knowledge-oriented, software components designed for human-robot interaction.

We focus on a specific class of interactions: collaborative task achievement supported by multi-modal and situated communication. Figure 1 illustrates this context: the two agents share a common space and exchange information through multiple modalities, and the robot is expected to achieve interactive object manipulation, fetch and carry tasks and other similar domestic tasks by taking into account, at every stage, the intentions, beliefs, perspectives, skills of the human partner. Namely, the robot must be able to recognise, understand and participate to communication situations, both explicit (*e.g.* the human addresses verbally the robot) and implicit (*e.g.* the human points to an object); the robot must be able to take part to joint actions, both pro-actively (by planning and proposing resulting plans to the human) and reactively; the robot must be able to move and act in a safe, efficient and legible way, taking into account social rules like proxemics.

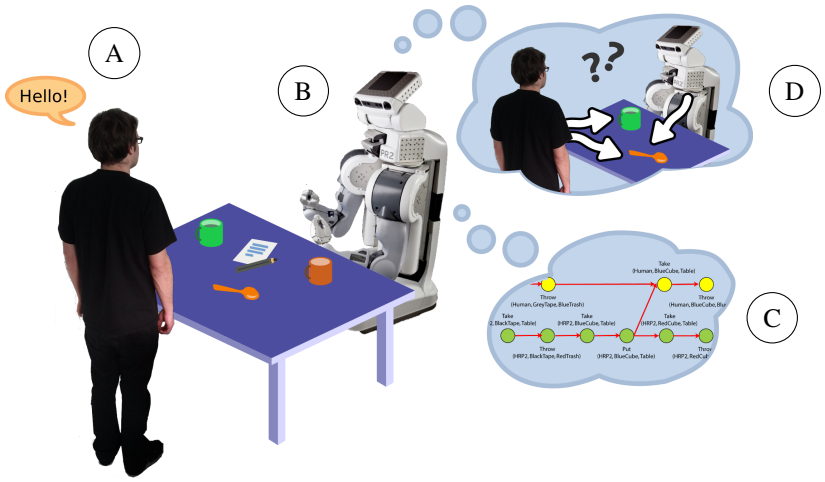


Figure 1: The robot reasons and acts in domestic interaction scenarios. The sources of information are multi-modal dialogue *(A)*, and perspective-aware monitoring of the environment and human activity *(B)*. The robot must adapt on-line its behaviours by merging computed plans *(C)* with reactive control. The robot explicitly reasons on the fact that it is (or not) observed by the human. Reasoning and planning take place at symbolic as well as geometric level and take into account agents beliefs, perspectives and capabilities *(D)* as estimated by the robot.

These three challenges, *Communication*, *Joint Actions*, *Human-Aware Execution*, structure the whole field of human-robot interaction research, and they can be analysed in terms of the cognitive skills that they require.

A *joint action*, for instance, builds from:

- a joint *goal*, which has been previously established and agreed upon (typically through dialogue),
- a physical environment, estimated through the robot's exteroceptive sensing capabilities, and complemented by inferences drawn from previous observations,
- a belief state that includes *a priori* common-sense knowledge and mental models of each of the interactors.

The robot controller (with the help of a task planner) decides what action to execute next, and who should perform it from the human or the robot (or both in case of a joint action), and finally controls or monitors its execution. The operation continues until the goal is achieved, is declared unachievable or is abandoned by the human [1].

This translates into several decisional, planning, representation skills that need to be available to the robot. It must be able 1. to represent and manipulate symbolic belief states, 2. to acquire and maintain them up-to-date with respect to state of the world and the task at hand, 3. to build and iteratively refine shared (human-robot) plans, 4. to instantiate and execute the actions it has to perform, and conversely, to monitor those achieved by its human partner.

Such abilities must be designed and implemented in a generic way and must provide several levels of parametrization such as they adapt to various environments, different tasks and variable levels of engagement of the robot, ranging from teammate behaviour to assistant or pro-active helper.

These are the challenges that we will discuss in this article.

1.2. Article organization

The remaining of the article discusses the robotic architecture we have built to tackle the AI challenges of human-robot interaction, and how its relates to other approaches. We propose to organise this discussion in four sections.

The next section introduces the architecture as a whole, as well as the knowledge model that we have developed for our robots. Then, section 3 gives some insights on each of the main components of the architecture, and tries to highlight their significance for artificial intelligence. Section 4 presents two studies that illustrate in a practical way what can be currently achieved with our robots. Section 5 finally restate our main contributions and discuss the key challenges that human-robot interaction brings to artificial intelligence.

2. Deliberative Architecture and Knowledge Model

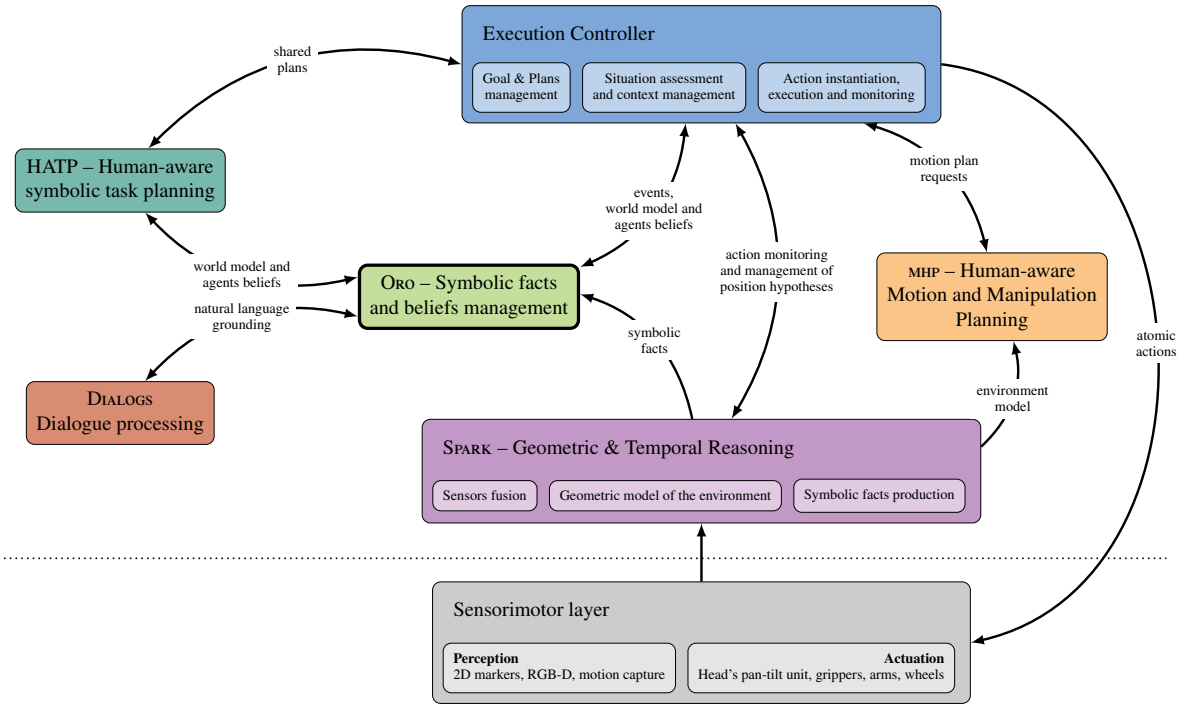


Figure 2: Overview of the architecture. A deliberative layer, composed of six main modules, interacts with a low-level sensori-motor layer. Knowledge is centrally managed in an active *semantic blackboard*, pictured above with a thick border. The links between components depicted on the figure underline the central role of the knowledge base: most of the datastreams are actually symbolic statements exchanged through this semantic blackboard.

2.1. Building a Human-Aware Deliberative Layer

Connecting multiple independent software modules in one coherent robotic architecture is not only a technical challenge, but also a design and architectural challenge.

In particular, dealing with the rich semantics of natural interaction with humans has been one major focus. To this end, we have been deploying over the last years explicit knowledge representation and manipulation in the deliberative layer of the robots as a *lingua franca* between our components.

Figure 2 gives an overview of our architecture. An active knowledge base (ORO), conveniently thought as a *semantic blackboard*, connects most of the modules together: the *geometric reasoning* module (SPARK) produces symbolic assertions (like ⟨ `BOOK1 isOn TABLE` ⟩) describing the state and dynamics of the robot's environment. These logical statements are stored in the knowledge base, and queried back by the language processing module (DIALOGS), the symbolic task planner (HATP) and the execution controller. The output of the language processing module and the activities started by the robot controller are likewise stored as symbolic statements.

For instance, when processing a sentence like "give me another book", the DIALOGS module queries the knowledge base: `find(?book type Book, ?book differentFrom BOOK1)`[1], and write back assertions like ⟨ `HUMAN desires GIVE_ACTION45, GIVE_ACTION45 actsOn BOOK2` ⟩ to ORO. The HATP planner then uses the knowledge base to initialise the planning domains with similar requests (`find(BOOK2 isAt ?location)`, etc.), and the execution controller typically monitor conditions (by subscribing to events like: `onNewMatch(HUMAN desires ?goal)`) and stores what the robot is currently doing (⟨ `myself currentlyPerforms GIVE_ACTION45` ⟩).

[1] We present study with a complete example at section 4.1.

To some extent, this architecture moves away from classical layered approaches found in robotics [2, 3, 4]. Interactions between components at the deliberative level are mostly bidirectional and we do not introduce layers of abstraction amongst software components[2]. Dialogue processing, for instance, illustrates this structure. This component does not purely act as an alternative sensing modality that would be fed to a distinct deliberative component. On the contrary, it actively queries the knowledge base to interpret and disambiguate natural language, and in that regards, it is fully integrated to the deliberative process itself (we explain natural language processing at section 3.3).

Our architecture however relates to *Beliefs, Desires, Intentions* (BDI) architectures. As put by Woolridge [5], BDI architectures are primarily focused on *practical reasoning*, *i.e.* the process of deciding, step by step, which action to perform to reach a goal. The management of the interaction between knowledge (the beliefs) and task and plan representation and execution (the desires and the intentions) is central, and aims at selecting at each step the best sub-goal. It becomes then an intention that the robot commits to.

As for any cognitive system, this fundamental interaction between knowledge and actions is central to our approach as well, and typically involves the dialogue module to acquire *desires* from the other agents, and the planner and the execution controller that to first decide to take or not into account an incoming desire as a *goal*, and then to generate and manage *intentions* from these goals through the symbolic task planner.

Building upon a BDI architecture, other activities are conducted in parallel, without being explicitly triggered by *desires* in the BDI sense, like situation assessment and monitoring or other forms of dialogue (including performative dialogue that can possibly change the internal state of the robot, but does not lead directly to the creation of *desires*, like assertion of new facts or question answering).

2.2. Knowledge Model

In our architecture, knowledge manipulation relies on a central server (the ORO server [6], Figure 5 top) which stores knowledge as it is produced by each of the other deliberative components (the clients). It exposes a `json`-based RPC API [7] to query the knowledge base. We represent knowledge as RDF triples in the OWL sub-language. Each time triples are added or removed from the knowledge base, a Description Logics reasoner (PELLET[3]) classifies the whole ontology and inserts all possible inferred triples. The clients of the ORO server are in charge of managing themselves the knowledge (when to add, when to update, when to retract knowledge): no meta-semantics is carried over that would let the server manage these dynamics itself.[4]

This architecture design (a central knowledge base that essentially appears as a passive component to the rest of the system – even though it actually actively processes the knowledge pool in the background, for instance with the reasoner) departs from other approaches like the CAST model [8] where knowledge is represented as a diffuse, pervasive resource, or the CRAM/KnowRob architecture [9] where the knowledge base is an active hub that proactively queries perceptual components to acquire knowledge. We believe that our design leads to a good *observability* (knowledge flows are explicit and easy to capture since they are centralised) as well as high modularity (modules communicate through an explicit and unified API).

At run-time, the knowledge available to the robot at a given time comes from three sources. *A priori* knowledge is stored in an ontology (the OPENROBOTS ontology, discussed below) and loaded at start-up. This static source covers the *common-sense* knowledge of the robot, and optionally some scenario-specific knowledge (for instance, about objects that are to be manipulated). The second part of the knowledge is acquired at run-time from perception, interaction and planning. The next sections go into details of these processes. Lastly, the third source of symbolic statements is the reasoner itself.

Contrary to other projects like KnowRob [10] that relies on the concept of *computables* to lazily evaluate/acquire symbolic facts when needed, we have an *explicit* approach where we greedily compute and assert symbolic statements (like spatial relations between objects, see Section 3.2). This design choice trades scalability for explicit reasoning: at any time, the full belief state is explicitly stated, and provides the reasoner with the largest possible inference domain. This is of special importance for an *event-based* architecture like ours (see Section 3.5), where, by definition, we do

[2] We do have lower-level modules to execute actions or manage sensors, but all cognition-related modules live in the same global deliberative layer.

[3] `http://clarkparsia.com/pellet/`

[4] There is one exception: so-called *memory profiles* let the server discard facts after a specific period of time. We present this experimental feature at section 3.1.3.

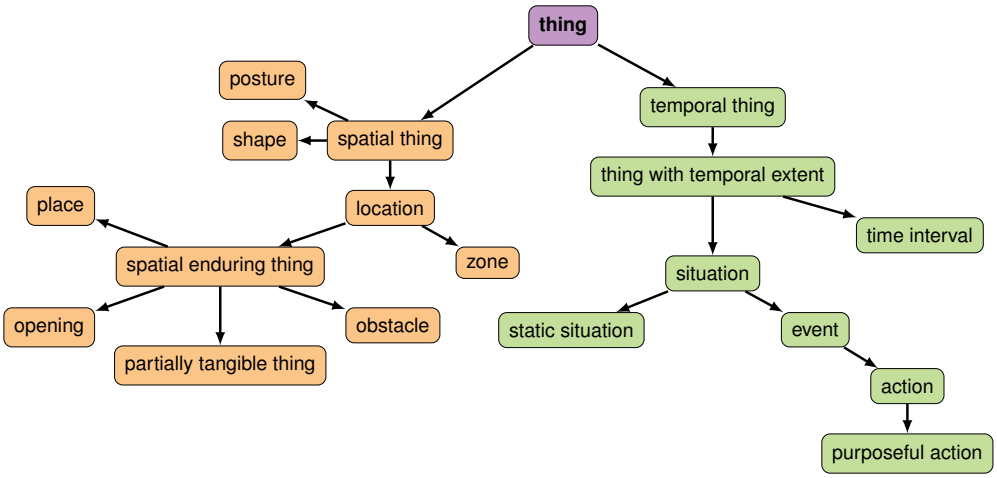


Figure 3: The upper part of the Oro common-sense TBox. All these concepts belong to the OpenCyc namespace.

not know beforehand what we are "looking for", and we need instead to greedily infer as much as possible to build a belief state as comprehensive as possible.

The OpenRobots Ontology. The Oro common-sense ontology has been designed from two requirements: covering our experimental needs and conforming as much as possible to the OpenCyc upper ontology.

This leads to a bidirectional design process: from *bottom-up* regarding the choices of concepts to model, *top-down* regarding the upper part of the taxonomy. This upper part of the ontology is pictured on Figure 3. All the classes visible in this figure belong to the OpenCyc namespace.

Aligning the upper part of the ontology on OpenCyc (as done by other knowledge representation systems, like KnowRob [10] or PEIS K&R [11]) has multiple advantages. First the design of this part of the ontology is generally difficult: it pertains to abstract concepts whose mutual relations comes to philosophical debates. The upper taxonomy of OpenCyc represents a relative consensus, at least within the semantic Web community. Then, because it is a well established project with numerous links to other on-line databases (like Wikipedia or WordNet), the reuse of key OpenCyc concepts ensures to a certain extent that the knowledge stored by the robot can be shared or extended with well-defined semantics. A good example is the concept of *Object*: in everyday conversation, an object is a relatively small physical thing, that can be typically manipulated. Normally, a human is not considered as an object. In Cyc, an object has a more precise definition: it is something *partially tangible*. This includes obviously the humans, and actually many other entities that would not be commonly said to be objects (the Earth for instance). Thus the importance of relying on well-defined and standard semantics to exchange informations between artificial systems in a non-ambiguous way.

Figure 3 also illustrates the fundamental disjunction in the Oro model between *temporal* and *spatial* entities (formally, $(\text{TemporalThing} \sqcap \text{SpatialThing})^{\mathcal{I}} = \emptyset$, with $\mathcal{I}$ the *interpretation* of our model).

The class `PurposefulAction` is the superset of all the actions that are purposefully performed by the robot (or another agent). Actual actions (*i.e.* subclasses of `PurposefulAction` like `Give` or `LookAt`) are not initially asserted in the common-sense ontology. They are instead added at run-time by the execution controller (in link with the symbolic task planner) and the natural language processor based on what the robot is actually able to perform and/or to interpret in the current context (*i.e.* the current robot configuration and the actions required by the scenario). The set of actions that the robot can interpret typically corresponds to the planning domain in use: the robot can only process the `GoTo` action if the corresponding task description with its pre- and post-conditions is available to the symbolic task planner.

The tree in Figure 3[5] is not equally developed in each directions at lower levels. The descendants of `PartiallyTangibleThing` (*i.e.* what we commonly call *objects*), for instance, are pictured in Figure 4. This excerpt from the ontology makes

[5]This subset of the ontology is indeed a tree. This has however not to be the case in general, and, as a matter of fact, the TBox of the whole Oro common-sense ontology does not form a tree

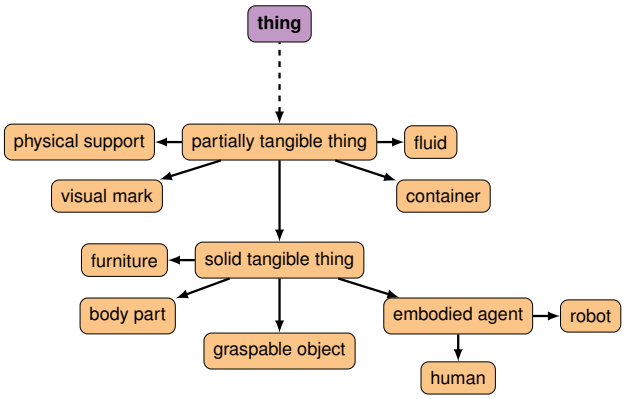


Figure 4: TBox of the specialisations of `PartiallyTangibleThing`.

the bottom-up design process clear: only few types of *partially tangible* things appear, and those are only subclasses relevant to the context of service robotics in an human-like environment. For performance reasons as well as clarity, we do not attempt to pro-actively extend the conceptual coverage of the field. We instead opportunistically extend the common-sense knowledge when required by experiments.

Lastly, the Oro common-sense ontology contains several rules and class expressions that encode non-trivial inferences. The definition of `Bottle` as found in the Oro ontology is a typical example:

$$\texttt{Bottle} \equiv \texttt{Container} \wedge \texttt{Tableware} \wedge \texttt{hasShape} \in \texttt{CylinderShape}^{\mathcal{I}} \wedge \texttt{hasMainDimension} \in [0.1, 0.3]$$

If a human informs the robot that a given object is indeed a bottle, the robot can then derive a few more informations on this object. And, conversely, if the human affirms that a car is a bottle, the robot may question this assertion because of the inconsistent size (the Dialogs module relies on such logical consistency checks when processing natural language inputs, both to ensure that the verbal input has been correctly acquired and to verify that what the human says does actually make sense).

We will come back and discuss in section 5.3 the strengths and weaknesses of this knowledge framework.

2.3. Symbol Grounding

Grounding (also called *anchoring* when specifically referring to the building of links between *percepts* and *physical objects* [12]) is the task consisting in building and maintaining a bi-directional link between sub-symbolic representations (sensors data, low-level actuation) and symbolic representations that can be manipulated and reasoned about [13]. This represents an important cognitive skill, in particular in the human-robot interaction context: in that case, the link that the robot has to established between its percepts and its symbol must *model* to a large extent the human representations in order to effectively support communication.

Symbol grounding connects hence the knowledge model to the perception and actuation capabilities of the robot, and our architecture adopts here also a bidirectional approach: *bottom-up* and *top-down*. The different components that we have mentioned so far provide a bottom-up grounding process: geometric reasoning and dialogue processing modules constantly build and push new symbolic contents about the world to the knowledge base where it becomes accessible to decisional layers. In parallel, the knowledge base relies on reasoning in a top-down way to produce new facts, *i.e.* new beliefs that may lead to actual decision making and physical actions.

3. Cognitive Skills

While the previous section provided an overview of our cognitive architecture as well as its associated knowledge model, this section discusses its main "building blocks". Each of these components may have connections to several others, and we invite the reader to refer to Figure 2 as a guide if needed.

In our context, we call *cognitive skills* the deliberative behaviours that are 1. *stateful* (keeping track of previous states is typically required for the component to perform adequately), 2. *amodal* in that the skill is not inherently bound to a specific perception of actuation modality, 3. manipulate *explicit semantics*, typically by the mean of symbolic reasoning, 4. operate at the *human-level*, *i.e.* are legible to the humans, typically by acting at similar levels of abstraction.

We start by discussing first the main *internal* cognitive capabilities, implemented in the knowledge base itself, and then discuss successively the situation assessment module SPARK, the dialogue processor DIALOGS, the symbolic task planner HATP, and finally the main features of our execution controllers.

3.1. Internal Cognitive Skills

We call *internal* those cognitive capabilities that are tightly bound to the knowledge model, and hence implemented directly within the ORO server. We present here three of them: *reasoning*, *theory of mind* modelling and our (naive) approach to *memory management*.

3.1.1. Symbolic Reasoning

As mentioned in the previous section, we use the Pellet open-source reasoner to reason on the knowledge base. This enables several standard inference services: *consistency checking*, *concept satisfiability*, *classification* and *realisation* (computation of the most specific classes that an individual belongs to). In case of logical inconsistency, the reasoner can also provide *explanations* that are used in particular for debugging.

Besides those, ORO server implements several algorithms (presented in [14]) to identify similarities and differences between concepts (classes or instances): the *Common Ancestors* algorithm, useful to determine the most specific class(es) that include a given set of individuals; the *First Different Ancestors* algorithm that returns what can be intuitively understood as the most generic types that *differentiate* two concepts; and *clarification* and *discrimination* algorithms that play a key role in the process of *interactive grounding* of the semantics of concepts (we discuss this process at section 3.3). Clarification and discrimination algorithms are based on what we call *descriptors*, *i.e.* properties of individuals, either statically asserted in the common-sense ontology, acquired by the robot through perception or pro-active questioning of the human partner, or derived from other reasoning algorithms like the *Common Ancestors* and *Different Ancestors*. The *discrimination* algorithm consists then in looking for discriminants, *i.e.* descriptors that allow a maximum discrimination among a set of individuals.

3.1.2. Theory of Mind

Theory of Mind (originally defined in [15]) is the cognitive ability that allows a subject to represent the mental state of another agent, possibly including knowledge that contradicts the subject's own model: for example, a book can be at the same time *visible* for myself, and *not visible* for you. Children develop this skill, which is essential to understand others' perspectives during interactions, around the age of three.

From a robotics point of view, it supposes the ability to build, store and retrieve separate models of the beliefs of the agents the robot interacts with. Our knowledge base implements such a mechanism: when the robots recognises that a new agent has been created in the knowledge base, it initialises a new, independent, ontology for this agent. All the ontologies that are created share the same common-sense knowledge, but rely on each agent's perspective for the actual instantiation: if the robot (geometrically) computes that the book is in its own field of view, but not in the human one, the robot knowledge contains the fact ⟨ `book isVisible true`⟩ while the human model contains ⟨ `book isVisible false`⟩ (this computation, called *perspective taking*, is discussed in the next section).

One classical application of this cognitive skill is the so-called *False-Belief* experiment (also known as the *Sally and Ann* experiment) [16]: a child is asked to watch a scene where two people, A and B, manipulate objects. Then A leaves and B hides away one object. When A comes back, we ask the child "where do you think A will look for the object?". Before acquiring a theory of mind, children are not able to separate their own (true) model of the world (where they know that the object was hidden) from the model of A, which contains *false beliefs* on the world (A still thinks the object is at its original position since he did not see B hiding it). Using separate knowledge models in the knowledge base, we have been able to replicate this experience with our robots [17].

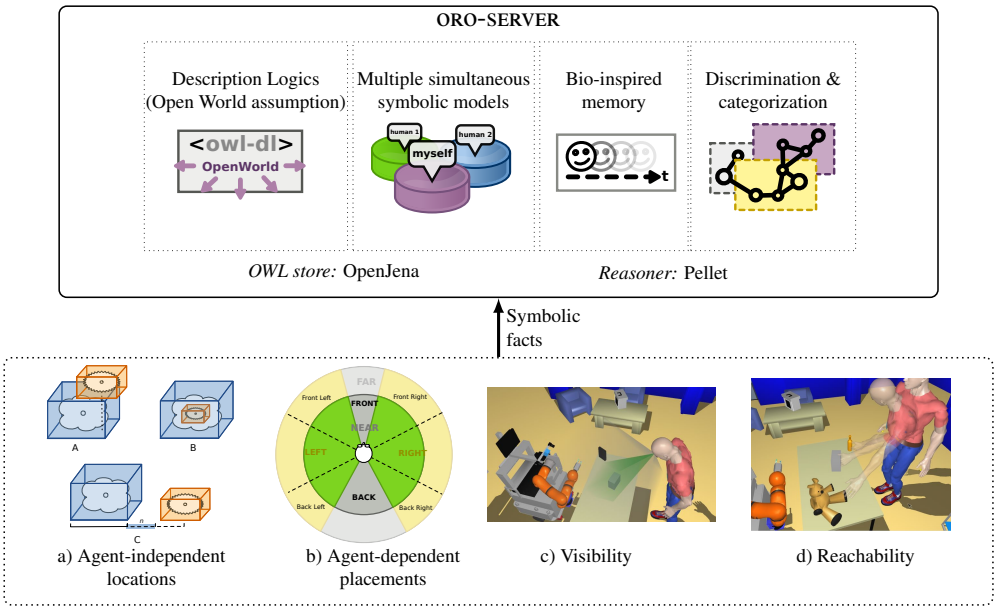


Figure 5: Functional overview of knowledge base (Oro server, top part) and the geometric situation assessment module (Spark, bottom part). Spark computes symbolic relationships between objects and agents, and feeds them to the knowledge base.

3.1.3. Working memory

Memory has been studied at length in the cognitive psychology and neuro-psychology communities: to give the main pointers, the idea of *short-term* and *long-term* memory is due to Atkinson and Shiffrin [18]; Anderson [19] proposes to split memory into *declarative* (explicit) and *procedural* (implicit) memories; Tulving [20] organises the concepts of *procedural*, *semantic* and *episodic* memories into a hierarchy. Short-term memory is eventually refined with the concept of *working memory* by Baddeley [21]. In the field of cognitive architectures, the Soar architecture [22] is one of those that tries to reproduce a human-like memory organisation. The GLAIR cognitive architecture [23] also has a concept of long term/short term and episodic/semantic memories.

It is worth emphasising that if memory is commonly associated to the process of forgetting facts after a variable amount of *time*, it actually covers more mechanisms that are relevant to robotics, like selective remembering triggered by a specific context or reinforcement learning.

The Oro server features a mechanism to mimic only minimalistic forms of memory families. When new statements are inserted in the knowledge base, a *memory profile* is optionally attached to them. Three such profiles are predefined: *short term*, *episodic* and *long term*. They are currently attached to different lifetime for the statements (respectively 10 seconds, 5 minutes and no time limit). After this duration, the statements are automatically retracted.

This approach is limited. In particular, *episodic* memory should primarily refer to the semantics of the statements (that is expected to be related to an event) and not to a specific lifespan.

We rely however on this mechanism in certain cases: for instance, some modules like the natural language processor use the *short term* memory profile to mark for a few seconds important concepts that are currently manipulated by the robot as *active concepts*: if a human asks the robot "Give me all red objects", the human, the `Give` action, and each red objects that are found are successively marked as *active concepts* by inserting statements such as ⟨ `human type ActiveConcept`⟩ in the short-term memory (which can be considered, in this case, to be a working memory). We use this feature to trace certain knowledge-related processes. On the other hand, our perception layer does not make use of this mechanism: as described in the next section, the environment model of the robot is continuously updated and the derived symbolic knowledge is therefore transient.

3.2. Acquiring and Anchoring Knowledge in the Physical World

Anchoring perceptions in a symbolic model requires perception abilities and their symbolic interpretation. We call *physical situation assessment* the cognitive skill that a robot exhibits when it assesses the nature and content of

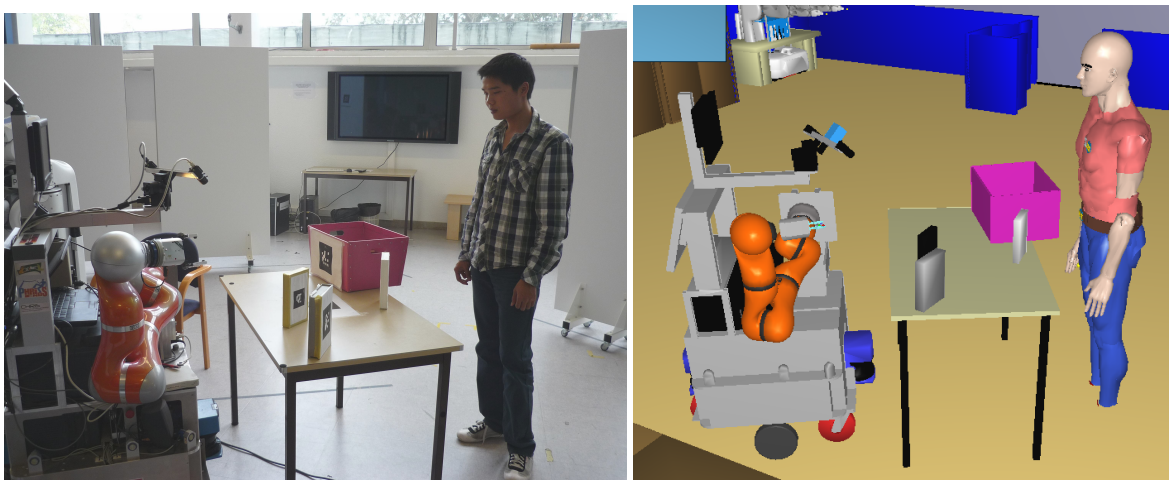
its surroundings and efficiently monitors its evolution.

Numerous approaches exist, like amodal (in the sense of modality-independent) *proxies* [24], grounded amodal representations [25], semantic maps [26, 27, 28] or affordance-based planning and object classification [29, 30].

We rely on a dedicated geometric and temporal reasoning module called SPARK (*SPAtial Reasoning & Knowledge*, [31]). It is a situation assessment reasoner that generates symbolic knowledge from the geometry of the environment with respect to relations between objects, robots and humans (Figures 5 and 6), also taking into account the different perspective that each agent has on the environment.

SPARK embeds an *amodal* [25] geometric model of the environment that serves both as basis for the fusion of the perception modalities and as bridge with the symbolic layer. This geometric model is made of CAD 3D models of the objects, furnitures and robots, and full body, rigged models of the humans, and it is updated at run-time by the robot based on its sensors (in most of our studies, objects are identified and localised through 2D fiducial markers, while humans are tracked with Kinect-like devices, optionally assisted by motion capture to accurately track the head motion, which is required to compute what the human is looking at).

SPARK is designed to run and continuously update the knowledge base (in our experiments, it runs at about 10Hz). At each step, it re-computes spatial relationships and affordances for the whole scene, and send the delta (new relations and relations that do not hold anymore) to the knowledge base. This approach may raise scalability concerns (we did not however hit major performance issues in our constrained scenarios), but greatly simplifies the management of the *dynamics* of the knowledge (*When do I discard outdated knowledge? When do I update it?*). Since it is equivalent to a reset of the reasoner domain, it also hides issues linked to non-monotonic reasoning (see [32] for a discussion on that question).



(a) Physical setting (b) Corresponding 3D model view

Figure 6: Toy scenario involving videotapes that have to be manipulated, with other objects used as supports and containers. After identification and localisation of the set of objects and acquisition of the position and posture of the human partner, the robot is able to compute that two tapes only are reachable by itself: the black and grey (in the 3D model) tapes. The third tape and the pink box are only reachable by the human. This situation leads to the symbolic model presented in Table 1.

3.2.1. Building an Agent-Aware Symbolic Model of the Environment

On Perspective Taking. *Visual* perspective taking refers to the ability for visually perceiving the environment from other's point of view. This ability allows us to identify the objects in situations where the visual perception of one person differs from the other one. In developmental psychology, one typical example consists of two similar objects in a room (*e.g.* two balls) where both are visible for the child, but only one is visible for the adult. Thus, when the adult asks the child to hand over "the ball", the child is able to correctly identify which ball the adult is referring to (*i.e.* the one visible from the adult point of view), without asking [33].

Robot's beliefs about itself (*robot's model*)	Robot's beliefs about the human (*human's model*)
⟨ PINK_BOX isReachable false⟩	⟨ PINK_BOX isReachable false⟩
⟨ WHITE_TAPE isReachable **false**⟩	⟨ WHITE_TAPE isReachable **true**⟩
⟨ BLACK_TAPE isReachable **true**⟩	⟨ BLACK_TAPE isReachable **false**⟩
⟨ GREY_TAPE isReachable **true**⟩	⟨ GREY_TAPE isReachable **false**⟩
⟨ WHITE_TAPE isVisible true⟩	⟨ WHITE_TAPE isVisible true⟩
⟨ BLACK_TAPE isVisible true⟩	⟨ BLACK_TAPE isVisible true⟩
⟨ GREY_TAPE isVisible true⟩	⟨ GREY_TAPE isVisible true⟩
⟨ WHITE_TAPE isOn TABLE⟩	⟨ WHITE_TAPE isOn TABLE⟩
⟨ BLACK_TAPE isOn TABLE⟩	⟨ BLACK_TAPE isOn TABLE⟩
⟨ GREY_TAPE isOn TABLE⟩	⟨ GREY_TAPE isOn TABLE⟩

Table 1: Symbolic facts computed from the situation depicted in Figure 6. Note how reachability differs for the two agents.

Besides, in order to compute a visual perspective, the actual visibility alone is not enough. We include not only what the other person sees in a given moment, but also what he *can* see with a minimal effort (moving the eyes or the head). To model the potential visibility of an object we compute a visibility ratio while turning the head of the agent model towards the object.

Spatial perspective taking refers to the qualitative spatial location of objects (or agents) with respect to a frame (*e.g. the keys on my left*). Based on this frame of reference, the description of an object varies [34]. Humans mix perspectives frequently during interaction. This is more effective than maintaining a consistent one, either because the (cognitive) cost of switching is lower than remaining with the same perspective, or if the cost is about the same, because the spatial situation may be more easily described from one perspective rather than another [35]. Ambiguities arise when one speaker refers to an object within a reference system (or changes the reference system, *i.e.* switches perspective) without informing his/her partner about it [36, 37]. For example, the speaker could ask for the "keys on the left". Since no reference system has been given, the listener would not know where exactly to look. However, asking for "the keys on your left" gives enough information to the listener to understand where the speaker is referring to. On the contrary, when using an exact, unambiguous term of reference to describe a location (*e.g.* . "go north") no ambiguity arises.

In SPARK, we use two types of the frames of reference: *ego-centric* (from the robot perspective) and *allo-centric*, *i.e.* addressee-centred (from the human perspective).

Symbolic Locations. Humans commonly refer to the positions of objects with symbolic descriptors (like *on*, *next to*...) instead of precise, absolute positions. These type of descriptors have been studied in the context of language grounding [38, 39, 40, 41, 42]. In SPARK we focus on agent-independent symbolic locations and agent-dependent, relative locations.

SPARK computes three main agent-independent relations based on the bounding box and centre of mass of the objects (Figure 5*a*): `isOn` holds when an object O_1 is on another object O_2, and is computed by evaluating the centre of mass of O_1 according to the bounding box of O_2. `isIn` evaluates if an object O_1 is inside another object O_2 based on their bounding boxes BB_{O_1} and BB_{O_2}. `isNextTo` indicates whether an object O_1 is next to another object O_2. We cannot use a simple distance threshold to determine if two objects are next to each other since the relation is highly dependent on the dimensions of the objects. For instance, the maximum distance between large objects (*e.g.* two houses) to consider them as being next to each other is much larger than the maximum distance we would consider for two small objects (*e.g.* two bottles). Thus, the relation between the dimensions and the distances of the objects are taken into account.

SPARK also compute symbolic facts related to agent independent world dynamics. The predicate `isMoving` states, for each tracked entity, whether it is currently moving or not.

Many topological relations are conversely dependent from the observation point. The predicate `hasRelativePosition` represents such spatial relations between agents and objects that are agent dependent. We compute these spatial locations by dividing the space around the referent (an agent) into n regions based on arbitrary angle values relative to

the referent orientation (Figure 5*b*). For example, for $n = 4$ we would have the space divided into *front, left, right* and *back*. Additionally, two proximity values, *near* and *far*, are also considered. The number of regions and proximity values can be chosen depending on the context where the interaction takes place.

Through perspective taking, SPARK computes for each agent a symbolic description of the relative positioning of objects in the environment. Table 2 summarises all the symbolic spatial relations computed by SPARK.

3.2.2. Building a Model of Agents

Building a grounded symbolic model of the physical environment does not suffice in general to fully ground the human-robot interaction, and a model of the current capabilities of the agents interacting with the robot is also required.

SPARK computes the following capabilities from the perspectives of each agent:

- *Sees*: this relation describes what the agent can see, *i.e.* what is within its field of view (FOV). In our current implementation, this affordance is computed by dynamically placing an OpenGL camera at the location of the eyes and running occlusion checks from it. In figure 5*c* the field of view of a person is illustrated with a grey cone (broader one). While he is able to see the two small boxes on the table in front of him, the big box on his right is out of his FOV, and therefore, he is not able to see it.

- *Looks At*: this relation corresponds to what the agent is focused on, *i.e.* where its focus of attention is directed. This model is based on a narrower field of view, the field of attention (FOA). Figure 5*c* shows the field of attention of a person with a green cone (narrower one). In this example only the grey box satisfies the `looksAt` relation.

- *Points At* holds when an object is pointed at by an agent. This relation is computed by placing a virtual camera on the hand, aligned with the forearm. `pointsAt` is particularly useful during interaction when one of the agents is referring to an object saying "this" or "that" while pointing at it.

  To make recognition more robust, these three first capabilities are filtered with an hysteresis function at the geometric level.

- *Reachable*: it allows the robot to estimate the agent's capability to reach an object, which is fundamental for task planning. For example, if the user asks the robot to give him/her an object, the robot must compute a transfer point where the user is able to get the object afterwards. Reachability is computed for each agents based on Generalised Inverse Kinematics and collision detection.

Table 2 lists all the symbolic facts that are computed by SPARK, along with the admissible classes for the subjects and objects of the statements.

Hypotheses on Objects States and Positions. It is sometimes difficult or even impossible to see and/or track an object in certain states. This happens, for instance, when the object is hidden in a container, when the robot holds it in its gripper, and more generally in any state in which it is hidden by something else. SPARK models the possible symbolic states of objects (whether the object is on a furniture, in an agent hand, in a container, etc.), and according to the robot perception of what has happened since the object was last seen, the robot builds a belief of the current possible state with associated probability model.

3.2.3. Primitive Action Recognition

Monitoring human activity is crucial for maintaining a coherent state of the world. While full human action and activity recognition is a task that requires knowledge and reasoning both on high level facts like goals, intentions and plans, as well as bottom-up data from agent and object motions, simple temporal and geometric reasoning on human hand trajectories and potential objects placements can provide clues for high level human monitoring processes. We call this temporal and geometric reasoning *primitive action recognition*. Those primitive actions are assessed through monitoring situations like "an empty hand is close to an object on a table" (precursor for a *pick*), or "a hand holding an object is over a container" (precursor for a *throw*). SPARK recognises a core set of such primitives that are relied upon by the robot to track the state transitions in the human plans.

Subject	Predicate	Object	*Notes*
Location	isAt ≡ cyc:objectFoundInLocation	Location	
	⟶ isOn ≡ cyc:above_Touching		
	⟶ isIn		
	⟶ isNextTo		
Location	isAbove ≡ cyc:above-Generally	Location	inverse of isBelow
			isOn ⟹ isAbove
Location	isBelow	Location	inverse of isAbove
Location	hasRelativePosition	Location	
	⟶ behind ≡ cyc:behind-Generally		inverse of inFrontOf
	⟶ inFrontOf ≡ cyc:inFrontOf-Generally		inverse of behind
	⟶ leftOf		inverse of rightOf
	⟶ rightOf		inverse of leftOf
Object	cyc:farFrom	Agent	
Object	cyc:near	Agent	
Agent	looksAt	SpatialThing	
Agent	sees	SpatialThing	
SpatialThing	isInFieldOfView	xsd:boolean	myself sees * ⇔ * isInFieldOfView true
Agent	pointsAt ≡ cyc:pointingToward	SpatialThing	
Agent	focusesOn	SpatialThing	looksAt ∧ pointsAt ⇒ focusesOn
Agent	seesWithHeadMovement	SpatialThing	
Agent	reaches	Object	

Table 2: List of statements describing agent-independent spatial relationships between objects (top), agent-dependent placements (middle) and attentional states and abilities of agents (bottom). "⟶" indicates sub-properties. When existing, the equivalent predicate in the OpenCyc standard (prefix cyc:) is specified. Note that some relationships are not computed by Spark, but are instead inferred by the reasoner.

3.2.4. Limitations

The main weakness of our current implementation relates to the way semantics are conveyed: because we use 2D markers for artifacts, we avoid issues related to object segmentation and recognition. Each object has a unique identifier, which enables us to load a proper CAD model suitable for 3D spatial reasoning and prevent recognition mistakes in the knowledge base. While Spark algorithms themselves are ignorant of the input sources, and would work equally well with a full object recognition stack, we do not have tackled it yet.

Besides, temporal reasoning (essential for accurate action recognition for instance) is not properly addressed in our stack. Temporal reasoning is used only locally, and does not allow for tracking of long sequences or global events.

3.3. Multi-Modal Communication

3.3.1. Natural Language Grounding

In our experience, natural language processing is one of the fields of human-robot interaction for which the introduction of the semantic layer has been most beneficial. In [43] we explain the techniques and the tool called Dialogs that we have developed for natural English language parsing and grounding, along with verbalisation and (minimalist) dialogue management.

Natural language input (in experiments, we rely on an Android-based interface, with Google speech recognition) is parsed into a grammatical structure, and atoms of each sentence are resolved with the help of the ontology to ground concepts like objects (*i.e.* when a user says "pick the can", it resolves to which instance of *can* the user is referring to) and actions. Sentences are sorted into questions, desires and statements, and processed accordingly. Figure 7 gives a example of the processing of a simple order.

The system supports quantification ("give me {a | the | some | all | any | ...} can"), thematic roles (action-specific predicates that qualify the actions), interactive disambiguation (the robot asks questions when it needs more information), anaphora resolution ("give *it* to me") based on dialogue history. It also permits knowledge extension by learning new semantic structures (for instance, a sentence like "learn that cats are animals" is converted into ⟨ Cat

Initial human knowledge	*Human input*
⟨ book_1 type Book⟩ ⟨ book_1 hasColor blue⟩	"Give me the book."

Generated partial statements	*Newly created statements*
⟨ ?obj type Book⟩ ⇒ ?obj = book_1	⟨ human desires sit_1⟩ ⟨ sit_1 type Give⟩ ⟨ sit_1 performedBy myself⟩ ⟨ sit_1 actsOnObject book_01⟩ ⟨ sit_1 receivedBy human⟩

Figure 7: Processing of a simple, non-ambiguous order, taken from [43]. Thematic roles (performedBy, actsOnObject, receivedBy) are automatically extracted by DIALOGS from the order "Give me the book.".

subClassOf Animal⟩ and added to the knowledge base after checking for possible contradictions with existing knowledge), interprets common temporal and place adverbs (like *above* or *tomorrow*) and translates to a certain extent *states* ("I'm tired") into *experiences* (⟨ HUMAN experiences state_1, state_1 hasFeature tired⟩). We kindly refer the readers to [43] for a complete discussion of these capabilities and the related algorithms.

3.3.2. Multi-Modality

Because all components rely on the same RDF formalism to format their outputs, the different communication modalities (*explicit* like verbal, deictic or based on gaze, or *implicit* like postures) are presented in a homogeneous way, as statements in the knowledge base. The dialogue grounding process makes use of them at two distinct levels to provide seamless multi-modal interaction.

First, particular steps of the grounding process explicitly check for the presence and value of specific facts: for instance, when several instances match a category (the human says "give me the bottle" and the robot knows about three bottles), the module may decide (it actually depends on the quantifier preceding the class, see the example at section 4.1) to discard some of them based on their *visibility* for the speaker (implicit communication context built on the human posture).

Another example, when the human says "this", the robot checks if the human is currently pointing at some object. In that case, *this* is replaced by the object focused on. Otherwise, the robot performs a lookup in the dialogue history to find a previous concept that the user could refer to (anaphora resolution).

Note that, while the system benefits from the complementary modalities, they are not all required. The dialogue system can run with the verbal modality alone, at the cost of a simpler interaction: if the human says "this" without the robot tracking what the human points at, no ⟨ HUMAN pointsAt ...⟩ fact is available in the knowledge base, and the robot fallbacks on the anaphora resolution step alone.

The second level of integration of multi-modality is implicit. By continuously computing symbolic properties from the geometric model, richer descriptions and hence discrimination possibilities are available. For instance, the robot may compute that one bottle is next to a glass while another one stands alone, and these symbolic descriptions are transparently re-used in a dialogue context to generate unambiguous references to discriminate between similar objects: "do you mean the bottle that is next to the glass?" ([14] provides a detailed account of our approach to interactive concept discrimination, along with the related algorithms).

3.4. Human-Aware Task Planning

Our execution controllers rely on symbolic task planning to convert long-term desires into a succession of atomic actions. We use in our architecture the HATP planner (*Human Aware Task Planner*) [44, 45, 46].

The HATP planning framework extends the traditional HTN (Hierarchical Task Network) planning domain representation and semantics by making them more suitable to elaborate plans which involve humans and robots acting together toward a joint goal. HATP is used by the robot to elaborate human-robot *shared plans* which are then used to anticipate human action, to propose to human to act or even to ask help from the human.

The HATP *planning domain* defines a set of methods describing how to decompose a task and represents the procedural knowledge of the robot as well as its knowledge about the actions that the human partner is able to achieve. It is stored outside of the central knowledge base, using a specific formalism (see the related discussion at the end of this section).

3.4.1. Agents and Action Streams

The originality of HATP resides in its ability to produce plans for the robot's actions *as well as* for the other participants (humans or robots), that we call *shared plans*.

An important feature of HATP is that it treats agents as "first-class entities" in the domain representation language. It can therefore distinguish between the different agents in the domain as well as between agents and the other entities such as tables and chairs. This facilitates a post-processing step in HATP that splits the final solution (sequence of actions) into two (or more if there are several humans) synchronised solution streams, one for the robot and one for the human, so that the streams may be executed in parallel and synchronised when necessary (Figure 8).

This effectively enriches the interaction capabilities of the robot by allowing for a form of human behaviour *prediction*, which is used by the robot execution controller to monitor the engagement of the human partner during joint tasks.

The planner also generates synchronisation points between the agents. In the plan depicted on Figure 8, the robot needs to wait for the success of the human agent's `PUT` action (estimated through the fulfilment of the action post-conditions, typically $\neg\langle$ `GREY_TAPE isOn TABLE`$\rangle$) to be able to start its own action.

We also use shared plans to verbalise the sequence of actions and to explain the human partner how a task may be shared [17].

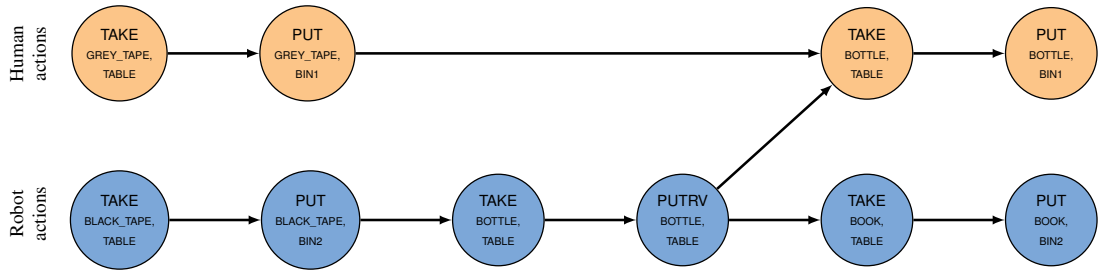


Figure 8: A plan produced by HATP with two action streams (human actions at the top, robot actions at the bottom, PUTRV stands here for *Put it so it is both reachable and visible*). The arrow between the two streams represents a synchronization point.

3.4.2. Action Costs and Social Rules

A cost and a duration function is associated to each action. The duration function provides a duration interval for the action achievement and is used, in one hand, to schedule the different streams and, in the other hand, as an additional cost function.

HATP includes mechanisms called *social rules* to filter plans so as to keep only those considered as suitable for human-robot interaction. The planner allows the specification of the following filtering criteria:

Wasted time: avoids plans where the human spends a lot of its time being idle,

Effort balancing: avoids plans where efforts are not fairly distributed among the agents taking part to the plan. It is indeed sometimes beneficial to balance efforts between the human and the robot,

Control of intricacy: avoids plans with too many interdependencies between the actions of agents mentioned in the plan, as a problem with executing just one of those actions could invalidate the entire plan. Also intricate human-robot activity may cause discomfort since the human will find himself repeatedly in a situation where his is waiting for the robot to act,

Undesirable sequences: avoids plans that violate specific user-defined sequences (for instance sequences which can be misinterpreted by the human).

Combining the above criteria, we typically yield shared plans with desirable interaction features like the human still being engaged in a number of tasks while its overall level of required efforts remains low, or avoiding having the

human to wait for the robot (by essentially preventing the action streams from having too many causal links between them).

Figure 9 illustrates such a socially-optimised plan where the *no wasted time* social rule is applied. The resulting shared plan is considered better than the one depicted on Figure 8 according to a global evaluation of the costs and rules.

In the current implementation, the social rules are effectively implemented by looking through all the plans produced and filtering out the ones that do not meet the desired requirements. In the future we intend to study algorithms that do such filtering online, rather than after initial solutions are found.

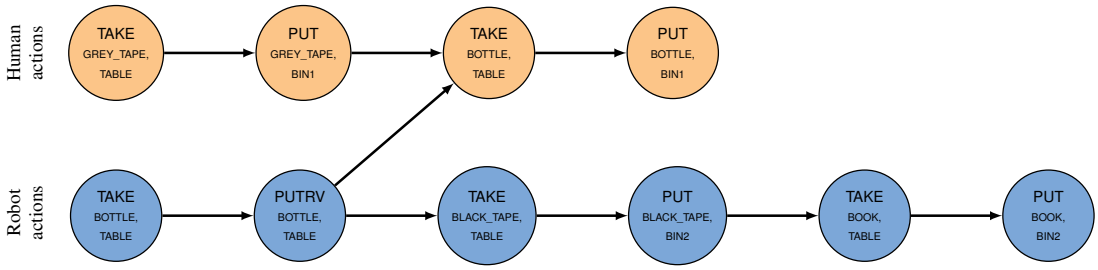


Figure 9: An alternative plan where the *no wasted time* social rule is used to optimise the total duration of the task.

Because HATP is a generic symbolic task planner, we have been able to design a planning domain at a semantic level which is close to the one used in the human-robot dialogue (the planner vocabulary contains concepts like `give`, `table`, `is on`...). This leads to a natural mapping (low "impedance mismatch") between the knowledge extracted from the situation assessment or the dialogue, and the planner.

Also, we would like to mention that, after some research we have decided against representing the planning domain (*i.e.* , the set of tasks with their pre- and post-conditions) in the knowledge base itself due to expressiveness issues (see appendix B of [47] for a detailed discussion). This effectively leads to independent declarative and procedural knowledge stores.

3.5. Robot Execution Control

While parts of the architecture (SPARK, ORO) have been deployed with external execution controllers (like CRAM [9] or the BERT platform [48], as reported in [6]), we have also developed dedicated robot controllers which integrate the whole stack introduced in Figure 2. SHARY [49] is the main one, written in the *Procedural Reasoning System* language [50]. We have also developed the Python-based PYROBOTS collection of tools to provide a large set of high-level actions and an event-based architecture well suited for prototyping. They both rely on extensive integration with the knowledge base, that serves as the primary source of semantics for decision making process.

One of the main role of SHARY is to control the elaboration and execution of shared plans [51, 52, 53]. This means essentially context-based refinement and robot actions execution, as well as monitoring of those achieved by its human partner [54]. The challenge is to build such abilities in a generic way, and to provide several levels of parametrisation allowing to adapt to various environments, and various levels of involvement of the robot ranging from teammate behaviour to assistant or proactive helper. Depending on this, the robot controller invokes the various human-aware planners and will react to events detected at ORO level (as described below).

SHARY's originality, as an execution control system, lies in its ability to take into account not only the task achievement but also communication and monitoring needed to support interactive task achievement [55, 56] in a flexible way. SHARY allows to link action execution *communication policies* in order to produce signals towards the human and to react to human actions. The *communication act* is the central concept in this formalism. It plays the role of transition condition, and, as such, represents an information exchange between the two agents. This exchange can be realised through dialogue, by an expressive motion or a combination of the two. It enables each agent to communicate their beliefs about the task to be achieved, in order to share mutual knowledge and to synchronise their activities. This is done through real-time task-based situation assessment achieved by the combination of SHARY monitoring processes and ORO inference mechanisms [57].

3.5.1. Desires and Experiences

We split the interaction situations stemming from the situation assessment and communication components in two categories: *desires* (related to *performative acts* in Austin's classification of speech acts [58]) and *experiences*.

Desires are typically human orders ("Give me that book"). The nature of the desired action (to pick, give, look, bring, show...), along with the action parametrization (thematic roles) are extracted from the knowledge base, and either passed to the task planner or executed if the procedure is directly available.

Experiences, on the other hand, comprise of emotions, states and questions (when asking a question, we consider the human to be in an *interrogative state*). When the knowledge base states that an agent *experiences* a particular emotion or state, the execution controller may decide to handle it, typically by trying to answer the question or using the emotional or physical state as a parameter for subsequent actions. As an example, when the speaker says "I feel tired", we change the motion planner parametrization to lower the effort the human needs to provide for the following joint manipulation tasks.[6]

3.5.2. Event-Driven Control

The ORO server proposes two paradigms to access its content: RPC-style queries (based on the standard SPARQL language) or events. A module can subscribe to an event by passing through an event pattern (in its simplest form, a partial statement like ⟨ `? type Book`⟩) and a callback. Each time a new instance of a book appears in the knowledge base, the callback is triggered.

This allows us to write reactive robot controllers with a high level of expressiveness: for instance, by subscribing to the event ⟨ `human1 desires ?action, ?action type Take, ?action actsOnObject ?obj, ?obj type Toy`⟩, we could trigger a behaviour when the human expresses (through dialogue, gestures...) that he wants a toy.

The robot controller designer does not need to directly care about how this *desire* is produced (this is delegated to perception modules), he can focus on the semantic of the desire.

Note also that we take advantage of the reasoning capabilities of the system: for example, the type of the object (⟨ `?obj type Toy`⟩) may not be explicitly asserted, but inferred by the reasoner based on other assertions.

3.5.3. Action Execution and Monitoring

Like most robotic architectures, actions are split into *atomic actions* that are combined into *tasks*. Tasks are created either statically or dynamically. The PYROBOTS controller, for instance, statically combines the actions listed in table 3 to implement the high-level tasks it exposes: `lookAt`, `moveTo`, `getFromHuman`, `showObject`, `giveObject`, `pickObject`, `bringObject`, `putObject`, `hideObject`. Our other controller, SHARY, relies on the symbolic task planner to dynamically generate, at run-time, suitable sets of atomic actions.

Manipulation
`attachobject`, `basicgive`, `basictake`, `close_gripper`, `configure_grippers`, `grab_gripper`, `handover`, `hide`, `open_gripper`, `pick`, `put`, `put_accessible`, `release_gripper`, `show`, `take`
Gaze control
`glance_to`, `look_at`, `sweep`, `switch_active_stereo_pair`, `track`, `cancel_track`
Navigation
`carry`, `follow`, `cancel_follow`, `goto`, `moveclose`, `waypoints`
Local navigation
`dock`, `rotate`, `translate`
Posture configuration
`extractpose`, `idle`, `manipose`, `movearm`, `rest`, `setpose`, `settorso`, `tuckedpose`

Table 3: Main PYROBOTS actions, sorted by categories. These actions are combined at run-time into higher-level *tasks*.

[6]Note that this specific example has been implemented as a proof-of-concept. A broader framework that would support action alteration based on the user's experienced states remains to be worked on.

Study	Focus	Reference
Point & Learn (2010)	Interactive knowledge acquisition	[6]
Spy Game (2010)	Interactive object discrimination	[14]
Interactive Grounding I (2011)	Multi-modal interaction, perspective taking	[62]
Roboscopie (2011)	Theater, reflection on the future of HRI	[63]
Cleaning the table (2011)	Whole stack integration	[64]
I'm in your shoes (2012)	False beliefs	[17]
Give me this (2012)	Natural joint object manipulation	[65]
Interactive Grounding II (2012)	Multi-modal interaction, perspective taking	[66]

Table 4: Main studies conducted with our cognitive architecture.

Manipulation and navigation actions listed in table 3 rely on a dedicated 3D motion planner (MHP, depicted in Figure 2): robot placement and end-effectors trajectories are computed on-line to allow object manipulation that take into account task specific constraints and human postures, abilities and preferences. We kindly refer the interested reader to [59, 60, 61] for details of these techniques.

4. Support Studies

Our architecture has been deployed and tested in a number of studies on several robotic platforms. Table 4 lists the most significant ones, with their main focuses and reference publications.

We briefly introduce two of them in order to illustrate in a practical way different aspects of the architecture. The first one is focused on knowledge representation and verbal interaction: the human asks for help to find and pack objects (*Interactive Grounding I* in Table 4). The second one (*Cleaning the table*) involves the SHARY execution controller and the HATP symbolic task planner. In this scenario, the human and the robot try to cooperatively remove objects from a table, and behaviours and motions are fully planned and then executed.

4.1. Interactive Grounding

This first study is based on the following scenario: Tom and Jerry are moving to London, so they are packing things in boxes. *Jido*, a mobile manipulator, is observing while they move things here and there (Figure 10(a)). They ask questions to Jido to know where certain objects are located. This study focuses on multi-modal, interactive grounding only, and the robot does not actually perform any action besides simple head movements.

Objects are perceived solely through 2D fiducial markers sticked on them, and humans are tracked through motion capture. The robot knowledge base is initialised with the ORO commonsense ontology. We next describe two situations where we can follow the internal robot's reasoning and the interaction with the user.

(a) Interactive grounding in a cluttered environment.

(b) Disambiguation through pointing.

Implicit Disambiguation Through Visual Perspective Taking. Tom enters the room while carrying a big box (Figure 10(a)). He approaches the table and asks Jido to hand him the video tape: "Jido, can you give me the video tape". The DIALOGS module processes this sentence, and queries the ontology to identify the object the human is referring to: `find(?obj type VideoTape)`.

There are two video tapes in the scene: one on the table, and another one inside the cardboard box. Thus, the knowledge base returns both: ⇒ `?obj = [BLACK_TAPE, WHITE_TAPE]`.

However, only one is visible for Tom (the one on the table). Although there is an ambiguity from the robot's perspective, the human referred to the tape using the definite article *the*: this is interpreted by the natural language processor as the human referring to a precise object, in that case, the visible one.[7]

Explicit Disambiguation Through Verbal Interaction and Gestures. In this second situation, Jerry enters the living room without knowing where Tom had placed the video tapes (Figure 10(b)). So he first asks Jido: "What's in the box?". Before the robot can answer the question it has to figure out which box Jerry is talking about. Similar to the previous situation, two boxes are visible: `find(?obj type Box)` ⇒ `?obj = [cardBoardBox, toolbox]`

However both are visible to the human and the previous ambiguity resolution procedure can not be applied. The robot generates a question (using the *Discrimination* algorithm that we mentioned at section 3.1.1) and asks Jerry which box he is referring to: "Which box, the toolbox or the cardboard box?" Jerry can answer the question, but he instead decides to point at it: "This box" (Figure 10(b)). SPARK identifies the `cardBoardBox` as being pointed at and looked at by the human and updates the ontology with this new information using a rule available in the commonsense ontology ⟨ `pointsAt(?ag, ?obj)` ∧ `looksAt(?ag, ?obj)` → `focusesOn(?ag, ?obj)`⟩. The DIALOGS module is then able to merge both sources of information, verbal ("this") and deictic to distinguish the box Jerry refers to:

⟨ `Jerry pointsAt carboardBox`⟩
⟨ `Jerry looksAt carboardBox`⟩
→ ⟨ `Jerry focusesAt carboardBox`⟩
⇒ `?obj = [cardBoardBox]`

Finally, the DIALOGS queries the ontology about the content of the box and the question can be answered: "Wall-E" (the label of the object, stored in the ontology, is used to interact with the user).

At this point Jerry wants to know where is the other tape: "And where is the other tape?". DIALOGS processes this sentence using the `differentFrom` OWL predicate:

⟨ `?obj type VideoTape`⟩
⟨ `?obj differentFrom WHITE_TAPE`⟩
⇒ `?obj = [BLACK_TAPE]`

Since there is only one possible "other" videotape, no specific disambiguation is required, and the robot verbalises the result of the query to the knowledge base ⟨ `BLACK_TAPE isOn table, BLACK_TAPE isNextTo toolbox`⟩ to "The other tape is on the table and next to the toolbox."

4.2. Cleaning the table *study*

This study (Figure 10) exhibits a richer decision-making process. The ORO server is used in conjunction with the HATP symbolic task planner and the SHARY execution controller to produce and execute a shared plan.

Figure 11 presents a practical sample of the whole task. It depicts a run with a single tape on a table. The tape is reachable by the robot only, while the bin where objects are supposed to be placed at the end is reachable by the human only: the robot needs to come up with a shared plan that involves joint actions.

The goal is first received by the execution controller (after processing of the user request by the DIALOGS module, not shown on the figure). At t_1 on Figure 11, the tape is computed by the robot as being reachable by the robot only (columns *Perception* and *Knowledge*), and the execution controller invokes the task planner, which produces a joint plan (column *Plan*) to move the tape so that the human can pick it and drop it into the bin.

[7] Other heuristics are available to the DIALOGS module: for instance, if a tape had been recently mentioned in the dialogue, this instance would have been selected instead as the referent.

Figure 10: Face-to-face setting of the *Clean the Table* scenario. The physical situation, the SPARK model, and the current step of the plan are visible on the picture.

The first task (`TAKE(GREY_TAPE, TABLE)`) is instantiated by first checking that the task pre-conditions hold, and then calling the 3D motion planner (column *Actions*, left). The motion planner returns two atomic actions (`PICK_GOTO` followed by `TAKE_TO_FREE`) that the controller executes (by first reaching the object, grasping it and bringing it back to a free position). The robot's perception monitors the evolution of the scene until the task's post-conditions are verified (at t_2, by satisfying the statement $\langle$ `ROBOT hasInHand TAPE`$\rangle$), and the next task is then started (placing the tape so that it becomes reachable by the human).

At t_3, the tape is now reachable by the human, and the next tasks (taking the tape and placing it in the bin) have to be performed by the human: the robot instructs the user to do so (verbal interaction not pictured on the figure) and monitors the actions of the human to detect when the tasks' post-conditions are satisfied (column *Actions*, right). When these post-conditions are fulfilled, the goal is considered to be completed.

This simple example illustrates how the symbolic facts are produced from the situation assessment, and, in parallel, used by the execution controller to assess the overall progress of the plan.

5. Discussion: When Artificial Intelligence Enables Human-Robot Interaction

This last section tries to clarify the "takeaway" message: what are the challenges that human-robot interaction brings to artificial intelligence? We first restate the challenge of *embodied cognition* in general and summarise what we see as the main decisional issues that need to be tackled to push forward human-robot interaction. Finally, we reflect on the importance of knowledge handling in robotic architectures that deal with complex semantics and put into perspective our choice of the RDF formalism.

5.1. Embodied Cognition

Robotics is traditionally regarded as the prototypical instance of *embodied* artificial intelligence, and this dimension is especially prevalent in human-robot interaction, where agents have to share and collaborate in a joint physical environment.

This leads to a tight coupling between the symbolic and the geometric realms: while AI at its origins was mostly a matter of symbolic models, it has been since recognised that not only the mind is not a purely abstract system, disconnected from the physical world, but even more, cognition fundamentally relies on its relation to the physical world (so-called *embodied cognition*). Varela [67] is one of the main discoverers of these mechanisms, and coined the concept of *enactivism* as the theoretical framework that study the links between cognition, embodiment and actions. This has since been thoroughly studied in robotics and artificial intelligence (Pfeifer and Bongard [68] is one of the reference).

The challenge of symbol grounding is also tightly linked to this issue. It corresponds to the identification or creation, and then, maintenance of a link between the symbol (the syntactic form of knowledge that the computer

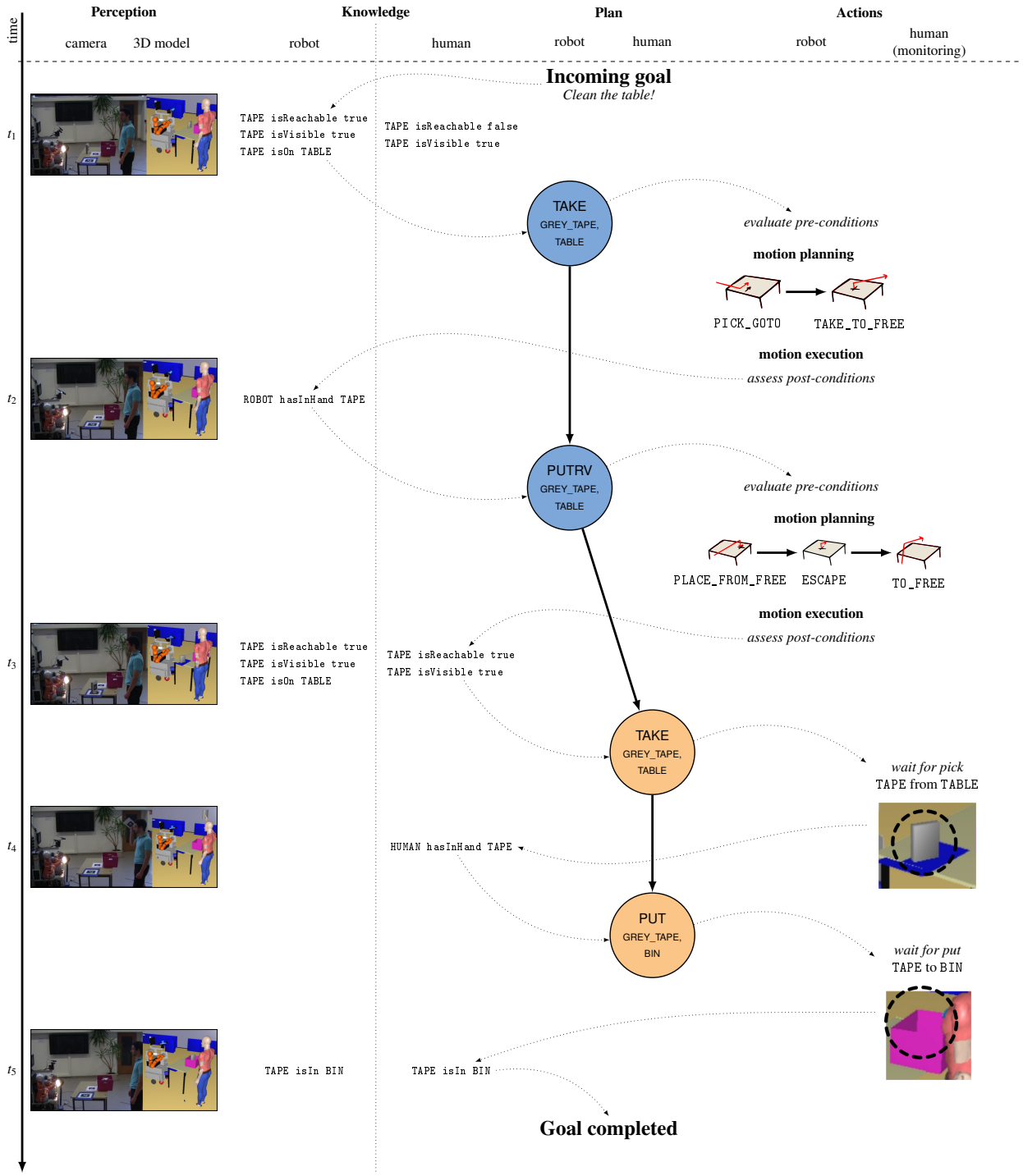


Figure 11: Chronology of the *Cleaning the table* scenario, presented here in a simplified version: a single object, `TAPE`, must be removed from the table and dropped in the bin `BIN`. The light arrows sketch the global execution flow.

manipulates) and its semantics, *i.e.* its meaning, anchored in the world (the relations between the symbol, the referent of the symbol, and mediating minds is classically referred as the *semantic triangle* and has been extensively studied in linguistics). The issue of grounding is well known in cognitive science and is summarised by Harnard [13] by this question: "how the semantic interpretation of a formal symbol system can be made intrinsic to the system?". This issue has a major practical importance in robotic: for a robot to be both endowed with a symbolic representational and reasoning system, and able to *act* in the physical world, it must ground its knowledge.

As we have seen, grounding is implemented at different levels in our architecture. The main source of grounded symbolic knowledge is the situation assessment module, SPARK. It builds symbolic facts from spatial reasoning, and also relies on (limited) temporal reasoning to track the world state and build explanations to interpret unexpected perceptions (like an object that suddendly disappears). Because SPARK also tracks humans, which enables perspective-aware spatial reasoning, it can produce as well grounded symbolic knowledge for the agents it interacts with. This is a typical *embodied* cognitive skill.

Grounding also occurs during verbal and non-verbal communication. The DIALOGS module grounds new concepts introduced by the speaker by asking questions until it can attach the new concept to concepts already present in the knowledge pool (typically, if I ask the robot "bring me a platypus", the robot asks "what is a platypus". I may answer "a mammal" and the robot will ask "what is a mammal?" and so on until it anchors the new concepts to ones it already knows). Embodied interactions (like gestures) are also taken into account at this level: we have presented how pointing, for instance, is used by the robot to ground *"this"* or *"that"* to the pointed artifact.

Note that because only objects marked with 2D fiducial markers are currently recognised (in our typical lab experiments, it may be a dozen of them), the robot's grounding mechanisms only work in a small-sized closed world. This assumption greatly facilitates the task, and we see that we have only scratched the surface of a generic grounding capability. Context, cultural background, "naive physics" knowledge, emotional state of the human are some of the numerous other determinants that need to be accounted for when grounding human-robot interaction.

5.2. The Key Decisional Issues

With this perspective on *embodied cognition* in mind, it seems appropriate to restate the key challenges brought by human-robot interaction to artificial intelligence.

5.2.1. The W-questions

Our context, where a robot has to achieve a task together with humans, raises issues to be tackled by the robots' decisional components, and that we can summarises as "W-questions": *What*, *Who*, *When*, *Where* and *How*?

What to do next, at different levels of abstractions and while taking into account not only the current state but also the long term goals, is the basic question for an intelligent robot. Its is made more complex here by having to deal with the partially observable physical and mental state of the human partner, and by the extended set of possible actions.

Who should act now is of key importance and also needs to be decided upon by the robot. It is sometimes expected as an intelligent behaviour for the robot to wait and let its human partner act instead. Correct management of *turn-taking* leads to various decisional challenges.

When to perform a given action is enriched here by the human-context since it has to take into account the human, his/her needs, his/her rhythms and pace, and his/her mental state. While performing its share of the task, the robot has to produce signals directed towards the human and to respond to signals produced back at the proper pace.

Where to perform an action plays an important role as well: the choice is not trivial and might need elaborated decision. The robot is expected to take into account effort sharing, visibility of its action by the human, disturbance or discomfort induced by its action.

How to perform an action, finally, needs to be reflected upon by the robot: several options to perform an action or to achieve a goal are often available, and selecting one is a non-trivial decision problem. Cost-based planners are one possible approach: they search for plans that satisfy an acceptable cost in terms of *acceptability* or *legibility*, for instance.

These five questions should not be considered independently from each others and often require, on the contrary, to be dealt with in a single decision step. The human-aware task and motion planners which we have built, are instances of systems which have been designed to deal with such intricate decision issues.

5.2.2. Putting the Humans into Equations

The correlate of these five *W-questions* is the issue the *human models*: taking appropriate decisions with and in presence of humans require appropriate models of the human. Models that tell the robot what the human *can do*, *would like to do*, *knows*, *could infer*, etc.

While the task of describing all the (dynamic!) human models that are useful to robots is immense (if doable at all), we claim that it is possible to devise and use such models in limited, but still interesting and useful, contexts such as collaborative human-robot objects manipulation, fetch-and-carry and associated activities in a domestic or professional environment.

In our implementation, perspective taking, for instance, is tightly connected to the symbolic knowledge models, and since our knowledge base allows for storage of one knowledge model per agent, we have been able to endow the robot with a simple theory of mind (as explained in section 3.1.2): we explicitly model what the robot knows about its partners in a symbolic way. This knowledge is then re-used in different places, to correctly interpret what the human says, or to plan tasks that are actually doable for the human.

The cognitive model that the robot builds for the agents it interacts with remains today simple and mostly focused on geometric features and affordances (*who sees what? what are our relative positions? what is reachable to whom?* etc.). Extending this knowledge with more subtle perceptions (emotional state for instance) remains to be done.

Motion and action execution also requires human models, and the one we use embeds human preferences and physical constraints that need to be accounted for when synthesizing robot motion or elaborating robot plans. This includes proxemics (human-robot distance) and associated issues (visibility) but also legibility and acceptability criterias expressed in terms of *social rules* that the elaborated plans should satisfy.

5.3. Knowledge and Robotics

As thorougly presented in this article, we have built the decisional capabilities of our robots around this idea of explicit knowledge manipulation. We finally briefely comment on this design choice, our implementation based on RDF, and the limitations that we perceive.

5.3.1. Knowledge and Architecture

As a whole, the components that we have introduced so far build a knowledge-oriented architecture: knowledge is explicitly stored in one central and consistent repository of facts, accessible for all modules. It relies on a strict formalism (OWL statements), with a well defined vocabulary (stated in the common-sense ontology). These first two points lead to a *loosely-coupled architecture* where modules can be removed or replaced easily by other ones as long as they share the same semantics (modules are defined by the knowledge they produce).

Also, we adopt a symbolic, reactive, event-driven approach to robot control. By managing events at the same level as the reasoner, we take full advantage of the inference abilities of Oro to trigger events whose `true` conditions can be (possibly indirectly) inferred.

And finally, this architecture allows for the combination of very different knowledge sources in a single homogeneous environment, bringing mutual benefits to components. For instance, the dialogue processing module can run without any situation assessment, but its disambiguation routines can transparently benefit from it when available (since richer symbolic descriptions of objects are then available).

Those items are not new *per se*. It is however interesting to underline the shift of focus this implies during the design and integration phases of robots: components of our deliberative layer are defined and glued together by the knowledge they produce and consume. Human-robot interaction, because it supposes operations at *human-level* and in environments with complex semantics, acts here as a motivational force.

5.3.2. RDF as a Formalism for Semantics

The Oro server relies on Description Logics (OWL) to represent and manipulate knowledge.

Relying on RDF triples and Description Logics has advantages such as good understanding of its trades-off, thanks to being widespread in the semantic Web community, the availability of mature libraries to manipulate the ontology, interoperability with several major on-line knowledge bases (OpenCyc, WordNet, DBPedia or RoboEarth [69] are some examples), open-world reasoning, and the formal guarantee of decidability (it is always possible to classify a Description Logics ontology).

It also has notable limitations, both fundamental (the suitability of Description Logics when reasoning on – typically non-monotonic– commonsense knowledge is questionable) and practical: RDF triples imply only binary predicates, which constrains the expressiveness of the system or leads to cumbersome reifications. Alternatives exist (like KnowRob [10]) that mix RDF with more expressive logic languages like Prolog with other limitations, like closed-world reasoning. Classification performance is another issue: from our experience, with an ontology sized for a standard experiment (about 100 classes and 200 instances), classification typically takes about 100ms, which becomes problematic during interactions. Besides, the performances are difficult to predict, since a seemingly inoffensive new statement may indirectly change abruptly the logical complexity of the whole knowledge model and lead to notable degradation of classification time.

This knowledge model also largely excludes representation of continuous phenomena (like time) or uncertain phenomena. When required (for instance for action recognition), these are managed by dedicated components (like Spark), and are not exposed at the semantic level.

While alternatives like *Answer Set Programming* have been successfully investigated in robotics [70, 71], in particular to deal with non-monotonic reasoning, we did not actually hit any brick wall while working with OWL ontologies. We may reconsider this choice at a later stage, but until now it has proven an effective framework to quickly explore implementations of new cognitive abilities (for instance, it has been conceptually and technically easy to add support for independent knowledge models, one per agent the robot interacts with).

Besides, because ontologies and RDF statements are relatively simple concepts to grasp, it also effectively helped to grow awareness amongst colleagues on the significance of the "semantic level" when developing new components for the robot.

5.3.3. Limits of Disambiguation at Semantic Level

Interaction with humans implies the ability to deal with semantics: semantics of verbal interaction, semantics of gestures, etc. As a consequence, it also implies to deal with semantic disambiguation.

One prototypical example of semantic disambiguation has been given in [14] with the child game *spygame*: two players are facing each other with a set of random objects in-between, one player mentally choose one object, and the other player has to guess the object by asking closed questions like *Is your object small or large?* Based on the knowledge it has acquired, the robot is able to minimise the number of questions required to find the object.

When playing this kind of game, however, the issue arises that the robot has no way to select which knowledge about the object is relevant in the interaction context. For instance, the knowledge base may store facts like ⟨ `obj1 type ActiveConcept`⟩ (which internally means that this concept was mentioned in a discussion in the last few seconds): this information is not a relevant property of `obj1` when trying to disambiguate concepts with humans. This distinction between *internal knowledge* (meaningful to the system only) and *common knowledge* (whose meaning is understood by all the interactors) has not been properly dealt with in our architecture.

Besides, even knowledge that belongs to the *common knowledge* may not be appropriate in a given interaction context. For instance, the system may compute that at a given instant the human is looking at the object: ⟨ `human looksAt obj1`⟩. This property makes sense to both parties, but in the context of the *spygame*, we would like to mainly use immanent properties, not volatile like a gaze. More research is required to identify relevant interaction contexts and knowledge classes attached to them.

5.4. The Next Steps

The design choices and the results presented here are still preliminary. While the general scheme we propose might be difficult to implement in a general sense, we believe that it is a reasonable challenge to implement it in the case of a personal robot assistant essentially devoted to fetch-and-carry, as well as for interactive manipulation tasks and associated activities.

One direction that we would like to further investigate is how to account for situations where divergent beliefs appear between the human and the robot. Preliminary results have been published in [17].

There is also extensive work to be done in order to refine the notion of "good shared plan" and "good/acceptable robot behaviour" in this context. There are obviously large avenues for learning and adaptation in this context.

Another broader direction to head deals with *contexts representation*. Contexts are currently often limited to the current spatial and temporal situation. Some models offer the possibility to jump in the past or to switch to another

agent's perspective, but in current approaches, selecting a context always basically consists in retrieving a set of beliefs corresponding to a situation, and temporarily replacing the current beliefs by those other ones. This misses the fact that at a given moment, not one but many contexts co-exist at different scales. We do not want to retrieve one monolithic set of beliefs, but instead carefully craft a context from several *atomic* contexts. Techniques for representation of overlapping pools of knowledge largely remain to be developed, as well as efficient algorithms to retrieve (or discard) such context-related pools of knowledge. This is a challenge not only for robotics, but more generally for artificial intelligence.

The ability to explicitly manage contexts and context switches would endow the robot with a cognitive capability similar to what is known as *context-dependent memory* in cognitive psychology. This is also related to Tulving's *autonoetic consciousness* [72]: the ability to reflect upon its own past or future experiences.

Much remain to be done to this regard, starting with a formal analysis of what are the relevant contexts for our robots.

Human-Robot Interaction is and is going to remain a challenging field for artificial intelligence. We hope that this contribution helps with clarifying some of these challenges.

Acknowledgements

Building such a robotic architecture is the work of many hands, and we would like to acknowledge here the worthwhile contributions of Samir Alili, Aurélie Clodic, Raquel Ros Espinoza, Mamoun Gharbi, Julien Guitton, Matthieu Herrb, Jim Mainprice and Amit Kumar Pandey.

This work has been supported by EU FP7 "SAPHARI" under grant agreement no. ICT-287513.

References

[1] G. Klein, J. M. Woods, D. D.and Bradshaw, P. J. Hoffman, R. R.and Feltovich, Ten challenges for making automation a "team player" in joint human-agent activity, IEEE Intelligent Systems 19 (6) (2004) 91–95.

[2] E. Gat, R. P. Bonnasso, R. Murphy, On three-layer architectures, Artificial intelligence and mobile robots (1998) 195–210.

[3] R. Volpe, I. Nesnas, T. Estlin, D. Mutz, R. Petras, H. Das, The CLARAty architecture for robotic autonomy, in: Aerospace Conference, 2001, IEEE Proceedings., Vol. 1, 2001, pp. 1/121–1/132 vol.1. doi:10.1109/AERO.2001.931701.

[4] D. Goldberg, V. Cicirello, M. B. Dias, R. Simmons, S. Smith, T. Smith, A. T. Stentz, A distributed layered architecture for mobile robot coordination: Application to space exploration, in: Proceedings of the 3rd International NASA Workshop on Planning and Scheduling for Space, 2002.

[5] M. Woolridge, Multiagent Systems. A Modern Approach to Distributed Artificial Intelligence., Massachusetts Institute of Technology, 1999, Ch. Intelligent Agents, pp. 27–78.

[6] S. Lemaignan, R. Ros, L. Mösenlechner, R. Alami, M. Beetz, ORO, a knowledge management platform for cognitive architectures in robotics, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2010.

[7] S. Lemaignan, Grounding the Interaction: Knowledge Management for Interactive Robots, 2012, Ch. The Knowledge API, pp. 161–174.

[8] N. Hawes, M. Zillich, J. Wyatt, BALT & CAST: Middleware for cognitive robotics, in: In Proceedings of the 16th IEEE International Symposium on Robot and Human Interactive Communication (ROMAN 2007), 2007, pp. 998–1003.

[9] M. Beetz, L. Mösenlechner, M. Tenorth, CRAM — A Cognitive Robot Abstract Machine for everyday manipulation in human environments, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2010.

[10] M. Tenorth, M. Beetz, KnowRob - knowledge processing for autonomous personal robots, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009.

[11] M. Daoutis, S. Coradeshi, A. Loutfi, Grounding commonsense knowledge in intelligent systems, Journal of Ambient Intelligence and Smart Environments (2009) 311–321.

[12] S. Coradeschi, A. Saffiotti, An introduction to the anchoring problem, Robotics and Autonomous Systems 43 (2-3) (2003) 85–96. doi:10.1016/S0921-8890(03)00021-6.

[13] S. Harnad, The symbol grounding problem, Phys. D 42 (1-3) (1990) 335–346. doi:10.1016/0167-2789(90)90087-6.

[14] R. Ros, S. Lemaignan, E. A. Sisbot, R. Alami, J. Steinwender, K. Hamann, F. Warneken, Which one? grounding the referent based on efficient human-robot interaction, in: 19th IEEE International Symposium in Robot and Human Interactive Communication, 2010.

[15] D. Premack, G. Woodruff, Does the chimpanzee have a theory of mind?, Behavioral and Brain sciences 1 (4) (1978) 515–526.

[16] A. Leslie, Theory of mind as a mechanism of selective attention, The new cognitive neurosciences (2000) 1235–1247.

[17] M. Warnier, J. Guitton, S. Lemaignan, R. Alami, When the robot puts itself in your shoes. managing and exploiting human and robot beliefs, in: Proceedings of the 21th IEEE International Symposium in Robot and Human Interactive Communication, 2012.

[18] R. Atkinson, R. Shiffrin, Human memory: A proposed system and its control processes, The psychology of learning and motivation: Advances in research and theory 2 (1968) 89–195.

[19] J. Anderson, Language, Memory, and Thought, Lawrence Erlbaum, 1976.

[20] E. Tulving, How many memory systems are there?, American Psychologist 40 (4) (1985) 385.

[21] A. Baddeley, Working memory, Current Biology 20 (4) (2010) R136–R140.

[22] J. Lehman, J. Laird, P. Rosenbloom, A gentle introduction to SOAR, an architecture for human cognition: 2006 update, Tech. rep., University of Michigan (2006).
[23] S. Shapiro, J. Bona, The GLAIR cognitive architecture, in: AAAI Fall Symposium Series, 2009.
URL http://aaai.org/ocs/index.php/FSS/FSS09/paper/view/948
[24] H. Jacobsson, N. Hawes, G.-J. Kruijff, J. Wyatt, Crossmodal content binding in information-processing architectures, in: Proceedings of the 3rd ACM/IEEE International Conference on Human Robot Interaction, ACM, New York, NY, USA, 2008, pp. 81–88. doi:10.1145/1349822.1349834.
[25] N. Mavridis, D. Roy, Grounded situation models for robots: Where words and percepts meet, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2006.
[26] A. Nüchter, J. Hertzberg, Towards semantic maps for mobile robots, Robotics and Autonomous Systems 56 (11) (2008) 915 – 926. doi:10.1016/j.robot.2008.08.001.
[27] C. Galindo, J. Fernández-Madrigal, J. González, A. Saffiotti, Robot task planning using semantic maps, Robotics and Autonomous Systems 56 (11) (2008) 955–966.
[28] N. Blodow, L. C. Goron, Z.-C. Marton, D. Pangercic, T. Rühr, M. Tenorth, M. Beetz, Autonomous semantic mapping for robots performing everyday manipulation tasks in kitchen environments, in: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), San Francisco, CA, USA, 2011.
[29] C. Lörken, J. Hertzberg, Grounding planning operators by affordances, in: International Conference on Cognitive Systems (CogSys), 2008, pp. 79–84.
[30] K. Varadarajan, M. Vincze, Ontological knowledge management framework for grasping and manipulation, in: IROS Workshop: Knowledge Representation for Autonomous Robots, 2011.
[31] E. Sisbot, R. Ros, R. Alami, Situation assessment for human-robot interaction, in: 20th IEEE International Symposium in Robot and Human Interactive Communication, 2011.
[32] J. McCarthy, From here to human-level AI, Artificial Intelligence.
[33] H. Moll, M. Tomasello, Level 1 perspective-taking at 24 months of age, British Journal of Developmental Psychology 24 (3) (2006) 603–614.
[34] L. Marin, E. A. Sisbot, R. Alami, Geometric tools for perspective taking for human-robot interaction, in: Mexican International Conference on Artificial Intelligence (MICAI 2008), Mexico City, Mexico, 2008.
[35] B. Tversky, P. Lee, S. Mainwaring, Why do speakers mix perspectives?, Spatial Cognition and Computation 1 (4) (1999) 399–412. doi:10.1023/A:1010091730257.
[36] C. Breazeal, M. Berlin, A. Brooks, J. Gray, A. Thomaz, Using perspective taking to learn from ambiguous demonstrations, Robotics and Autonomous Systems (2006) 385–393.
[37] R. Ros, E. A. Sisbot, R. Alami, J. Steinwender, K. Hamann, F. Warneken, Solving ambiguities with perspective taking, in: 5th ACM/IEEE International Conference on Human-Robot Interaction, 2010.
[38] J. O'Keefe, The Spatial Prepositions, MIT Press, 1999.
[39] C. Matuszek, D. Fox, K. Koscher, Following directions using statistical machine translation, in: Proceedings of the International Conference on Human-Robot Interaction, ACM Press, 2010.
[40] T. Regier, L. Carlson, Grounding spatial language in perception: An empirical and computational investgation, Journal of Experimental Psychology.
[41] J. Kelleher, G. Kruijff, Incremental generation of spatial referring expressions in situated dialog, in: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics, Association for Computational Linguistics, 2006, pp. 1041–1048.
[42] S. N. Blisard, Modeling spatial referencing language for human-robot interaction, in: in Proc. IEEE Intl. Workshop on Robot and Human Interactive Communication, 2005, pp. 698–703.
[43] S. Lemaignan, R. Ros, E. A. Sisbot, R. Alami, M. Beetz, Grounding the interaction: Anchoring situated discourse in everyday human-robot interaction, International Journal of Social Robotics (2011) 1–19doi:10.1007/s12369-011-0123-x.
[44] S. Alili, V. Montreuil, R. Alami, HATP task planer for social behavior control in autonomous robotic systems for HRI, in: The 9th International Symposium on Distributed Autonomous Robotic Systems, 2008.
[45] S. Alili, M. Warnier, M. Ali, R. Alami, Planning and plan-execution for human-robot cooperative task achievement, in: 19th International Conference on Automated Planning and Scheduling, 2009.
[46] R. Lallement, L. De Silva, R. Alami, Hatp: An htn planner for robotics, in: Proceedings of the PlanRob 2014, ICAPS, 2014.
[47] S. Lemaignan, Grounding the interaction: Knowledge management for interactive robots, Ph.D. thesis, CNRS - Laboratoire d'Analyse et d'Architecture des Systèmes, Technische Universität München - Intelligent Autonomous Systems lab (2012).
[48] S. Lallée, S. Lemaignan, A. Lenz, C. Melhuish, L. Natale, S. Skachek, T. van Der Zant, F. Warneken, P. Dominey, Towards a platform-independent cooperative human-robot interaction system: I. perception, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2010, pp. 4444–4451. doi:10.1109/IROS.2010.5652697.
[49] A. Clodic, H. Cao, S. Alili, V. Montreuil, R. Alami, R. Chatila, SHARY: a supervision system adapted to human-robot interaction, in: 11th International Symposium on Experimental Robotics, 2008.
[50] F. Ingrand, R. Chatila, R. Alami, F. Robert, PRS: A high level supervision and control language for autonomous mobile robots, in: Proceedings of the IEEE International Conference on Robotics and Automation, Vol. 1, 1996, pp. 43–49.
[51] B. J. Grosz, S. Kraus, Collaborative plans for complex group action, Artificial Intelligence 86 (1996) 269–358.
[52] H. H. Clark, Using Language, Cambridge University Press, 1996.
[53] C. Kemp, E. Edsinger, A. Torres-Jara, Challenges for robot manipulation in human environments, Robotics & Automation Magazine.
[54] C. Breazeal, Towards sociable robots, Robotics and Autonomous Systems (2003) 167–175.
[55] C. Rich, C. L. Sidner, Collagen: When agents collaborate with people, Proceedings of the first international conference on Autonomous Agents.
[56] C. L. Sidner, C. Lee, C. Kidd, N. Lesh, C. Rich, Explorations in engagement for humans and robots, Artificial Intelligence 166 (1-2) (2005)

140–164.
[57] M. Fiore, A. Clodic, R. Alami, On planning and task achievement modalities for human-robot collaboration, in: 14th International Symposium on Experimental Robotics, 2014.
[58] J. Austin, J. Urmson, M. Sbisà, How to do things with words, Harvard University Press, 1962.
[59] E. A. Sisbot, A. Clodic, R. Alami, M. Ransan, Supervision and motion planning for a mobile manipulator interacting with humans, in: Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction, 2008.
[60] J. Mainprice, E. A. Sisbot, L. Jaillet, J. Cortes, R. Alami, T. Simeon, Planning human-aware motions using a sampling-based costmap planner, in: IEEE International Conference on Robotics and Automation, 2011.
[61] A. Pandey, M. Ali, M. Warnier, R. Alami, Towards multi-state visuo-spatial reasoning based proactive human-robot interaction, in: Proceedings of the 15th International Conference on Advanced Robotics, IEEE, 2011, pp. 143–149.
[62] S. Lemaignan, R. Ros, R. Alami, M. Beetz, What are you talking about? grounding dialogue in a perspective-aware robotic architecture, in: Proceedings of the 20th IEEE International Symposium in Robot and Human Interactive Communication, 2011.
[63] S. Lemaignan, M. Gharbi, J. Mainprice, M. Herrb, R. Alami, Roboscopie: A theatre performance for a human and a robot, in: Proceedings of the 2012 Human-Robot Interaction Conference, 2012.
[64] R. Alami, M. Warnier, J. Guitton, S. Lemaignan, E. A. Sisbot, When the robot considers the human..., in: Proceedings of the 15th International Symposium on Robotics Research, 2011, to appear.
[65] M. Gharbi, S. Lemaignan, J. Mainprice, R. Alami, Natural interaction for object hand-over, 2013.
[66] S. Lemaignan, R. Alami, Talking to my robot: from knowledge grounding to dialogue processing, in: Proceedings of the 2013 Human-Robot Interaction Conference, 2013.
[67] F. Varela, E. Thompson, E. Rosch, The embodied mind: Cognitive science and human experience, The MIT Press, 1992.
[68] R. Pfeifer, J. Bongard, How the body shapes the way we think: a new view of intelligence, MIT press, 2007.
[69] M. Waibel, M. Beetz, J. Civera, R. D'Andrea, J. Elfring, D. Galvez-Lopez, K. Haussermann, R. Janssen, J. Montiel, A. Perzylo, et al., Roboearth, Robotics & Automation Magazine 18 (2) (2011) 69–82.
[70] X. Chen, J. Ji, J. Jiang, G. Jin, F. Wang, J. Xie, Developing high-level cognitive functions for service robots, in: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems, AAMAS '10, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 2010, pp. 989–996.
[71] E. Erdem, E. Aker, V. Patoglu, Answer set programming for collaborative housekeeping robotics: representation, reasoning, and execution, Intelligent Service Robotics 5 (4) (2012) 275–291.
[72] E. Tulving, Memory and consciousness, Canadian Psychology/Psychologie Canadienne 26 (1) (1985) 1.