Programming a humanoid robot in natural language:
an experiment with description logics

Nicola Vitucci , Alessio Mauro Franchi, Giuseppina Gini
DEIB, Politecnico di Milano
Milano, Italy

Abstract— **Programming a robot in natural language is a long time dream. Today applications, using a simplified vocabulary, are still far from really interacting in natural language with a robot to ask it to perform actions. Here we investigate why the task is still difficult, focusing on the embodied nature of language. To improve reasoning on we propose using description logics as a way to represent sensory motor information, and conclude with a case study using description logics to ask the robot for every day actions.**

Keywords—description logic; embodiment; natural language

I. INTRODUCTION

Traditionally the aim of robot programming languages has been to create robotic applications. Often robotic applications are a mix of robot commands, drag-n-drop interfaces, and point-n-click commands. Various programming languages are available to create robot applications. At run time unexpected situations made the system to stop or to try some reasoning to overcome the problems [11].

Cognitive robotics aims to design systems able to act out of structured environments, by mimicking human cognitive abilities such as perception, memory and reasoning. Many disciplines such as neuroscience, psychology, cognitive science, linguistics and computer science are involved for producing reliable and efficient models of cognition. It is clear that a fully cognitive system is necessary to be able to program the robot actions through a dialogue in natural language.

For humanoid robots, which are expected to cooperate with humans, a vocal input has been proposed. For instance, in [8] natural language sentences are translated into some robot behaviors to move the robot in a cooperative task with humans.

Often natural language in robotics has been seen as a separate problem, which aimed at obtaining a more direct interaction between man and machine. However, language is not only a powerful mean of communication: it is in fact deeply linked to the inner organization of the mind and it guides its development. Alan Turing in his test "Imitation Game" concluded that language manipulation is a necessary condition for a machine to be intelligent.

Nowadays all the applications of NLP in AI are getting more and more efficient, but they miss an important factor; linguistic skills do not cooperate with all the other cognitive mechanisms of the artificial agent but are considered as a stand alone module, able to read and translate commands; it is a simple human-robot interface and does not contribute to the development of the mind of the robot, in contrast to natural mechanisms.

To create in artificial agents such an ability, natural language and robots perceptions should be linked together; the comprehension of natural language by robots should be based on sensory-motor experiences and not on a sort of hard coded semantic [7]. In a longer time perspective, a language based on the experience of the robot would allow it to autonomously extract knowledge about the environment, integrating this information with data from its own actions and related sensorial feedbacks.

An important problem in robotics is the symbol-grounding problem [13] that concerns the relation between words or symbols with their meaning. An interesting study showed that sensory-motor integration could improve symbol-grounding processes [18]. The stimuli that human beings receive come from different sources, thus, in order to replicate our abilities to make sense of such a wealth of information, robots should be able to integrate them. Embodied robotics uses this finding.

However the integration of symbolic reasoning with sensory motor models is still open. Today robotics is approaching more and more the problem of knowledge representation and reuse to improve the capabilities of robot learning and social interactions. While the robotics problems are still dominated by unpredictable interactions with the real world, the societal robot is expected to interact with people sharing their reasoning and beliefs. Semantic technologies have seen an up-rise in robotics, as indicated by the creation of the Autonomous Robots Ontology Subgroup of IEEE RAS, and also by large projects, as KnowRob and RoboHow[1].

The use of natural language in robotics involves two sides: how the robot masters the language to create and organize its knowledge, and how the robot understands human language to receive orders. While we have worked in the past on the first point [10], here we focus on the second topic. In particular we explore the use of Description Logics (DLs) in a cognitive architecture as a way to extend data base search through reasoning. We discuss about assumptions that should hold in robotics for creating semantic representations. Then we use an ontology to extract robot commands from natural language. The extracted meaning is then matched against the cognitive structure of the robot that is organized in behaviors.

---

[1] knowrob.org/ and robohow.eu/

Our examples are taken considering the test for the iCub robot developed in the POETICON and POETICON++ projects[2]: stir the coffee. Stir the coffee is a simple verbal request for people; when addressed to a robot it will call for planning (finding a spoon, inserting it in the cup, rotating), for a complex motor program (move the arm so the spoon tip slowly circles in a constrained space), and for fully perceptual and haptic capabilities. State of the art learning and imitation learning techniques are adequate for teaching a robot the corresponding behavior, but how to go beyond the learned motor program for the specific behavior? Could the robot generalize this behavior and be able, for instance, to stir a soup? And what tool will be selected in this case? We show how our system is able to make this kind of generalization.

II. SEMANTIC TECHNOLOGIES AND ROBOTICS

A. Sematic technologies

The renewed interest in using symbolic systems in robotics has been driven by the growing applications of semantic web technologies and ontologies. Ontologies have emerged as a way to provide a structured representation of knowledge. They are deterministic in nature, consisting of concepts and facts about a domain and their relationships.

Description logics (DL)[4] define a logic-based framework for knowledge representation and reasoning, complemented by semantic query languages, which constitute the semantic counterpart of query languages for databases. Since their introduction in the 1980s DLs have seen a large research effort to make them expressive and decidable. DLs are used for building ontologies and knowledge bases, which are collections of statements regarding *concepts* (also called classes), *roles* (also known as properties) and *individuals* (denoted also as instances or objects) organized in a Terminological Box (TBox), containing axioms describing concepts and their relations, and an Assertional Box (ABox), containing axioms describing individuals and their relations with concepts and other individuals. Such statements can be represented as triples (subject-predicate-object) and stored in relational databases, queried using languages as SPARQL [4].

TBox axioms, that usually define a hierarchy of concepts, can be used also as consistency checks: if tables need to have four legs and a specific instance of a table has five, the knowledge base is inconsistent. Additionally, TBox axioms can be used to infer general knowledge related to a whole class: for instance, if the class of stable objects is defined as the class of "objects having at least three legs", tables described with four legs will be inferred as stable, and this will hold for all the instances of the class.

On the other hand, instances can be used like entries in a database: ABox queries can be executed to retrieve from the knowledge base objects having the attributes of interest. For instance, if we are looking for objects having four long square-based legs and a square top, the instance table will be retrieved; anyway, when performing this kind of queries, one has to be aware of the Open World Assumption of DL which can give unexpected results.

A common choice in robotics is to use concepts to represent classes of objects and individuals to model single objects. Complex concepts like *a table has four legs and a surface* and *a table is either made of wood or of plastic* can be formed using intersection, union and other available constructs. It is important to note that concepts can only have a tree-like representation in OWL, which means that axioms as *a table has four legs parallel to each other* cannot be expressed. More in general, it is difficult to write a TBox formalization if it has to include parts, topology axioms, constraints using reflexive and transitive roles and so on; in the robotic domain this can be a concern when attempting to model places, objects or kinematic chains as classes.

Imagine to model red objects. What differences the three possible formulations *Red, hasColor.Red hasColor.{red}*? A class Red of red objects can be convenient when there are many entities that need to be partitioned according to this characteristics, otherwise it might be more convenient to use an attribute, like color, and a value, such as red. From an implementation point of view, the first form requires the use of a class Red to which red-colored objects belong; the second form requires a property has-Color having objects as its domain and a color as its range, thus an object would have a property linking it to another object which is the color (in this case belonging to the subclass Red) and requiring the full existential quantification construct; the third form would link the object to a specific instance red (possibly belonging to the Color class) entitled to represent the red color itself, requiring a specific construct $\mathcal{O}$ for enumerated classes.

The current standard language for building ontologies is OWL 2. Sometimes it is not necessary to make use of the full OWL 2, so to optimize the performance some fragments, called *profiles,* have been created: EL for large TBoxes, QL for large ABoxes, and RL as a compromise. They differ in the constructs (intersection, union, negation, inverse, existential quantifier) that can be used within a subsumption axiom of the kind $C \subseteq D$. The right level of expressivity of a DL has to be assessed having in mind the final application, because it restricts the choice of reasoning engines and of storage systems that can be used. When robot ontologies are used to represent general world and action knowledge, their connection to real experimental data and low-level sensor data is still minimal. In fact sensor data are usually stochastic, and are better interpreted in a probabilistic framework. We have proposed a solution in [23, 24].

B. Robot cognitive architectures

Several cognitive architectures have been developed, the oldest being SOAR [15], a symbolic architecture extended to include sub-symbolic processing, and ACT-R [1], which has its roots in cognitive psychology. There are several problems in adopting such general architectures in robotics, from generating the right symbols to using low-level information.

Cognitive architectures can be enriched with a semantic memory, which can be considered as a long-term concept-based memory containing general knowledge. Often this

[2] http://www.poeticon.eu

module makes only use of lexical information, meaning that it does not provide any grounding or reference to the real objects they refer to but rather to their descriptions. Common knowledge resources such as Cyc [16] or ConceptNet [17] are respectively too broad or lacking a formal model, thus are not actual candidates for robotics problems.

From a cognitive point of view it is not useful to rely only on symbolic models on the one hand or statistical models on the other. The problem becomes how to derive knowledge directly from perception and how to organizing the conceptual system. In perceptual symbol systems [5] perception through sensory-motor systems is stored as patterns of activations in the brain, so that perceptual symbols associated to single perceptual components (such as color and shape) can be later retrieved and organized as concepts in a conceptual system. In this case perceptual information is not just "attached" to symbols by means of a translation in a different language, because this would bring back the symbol-grounding problem [13]; on the contrary, symbols are amodal and represent subsets of a perceptual state.

In order to be effectively usable, such representation should not only record perceptual information but also interpret it to distinguish specific instances (instances) from general categories (classes). Perceptual symbols are not immutable as they can change over time; as they are componential rather than holistic, in the sense that they regard "parts" of a perception, they are intrinsically qualitative and can represent classes of such entities. Perceptual symbols are multimodal, including all the sensory channels, introspection, and proprioception. They offer a different approach to categorization: an instance is decided to belong to a category if it can be simulated by the concept related to such category using its own representation. In [5] the decision on whether a real entity can be categorized by a certain concept is not taken on a logical basis but by comparing low-level representations of perceptions, and linguistic symbols are used to index and control simulations.

A consequence is the need to find out which features can be derived from perception and used as symbols; such features should be nameable in order to be communicated, but they should not be decided "a priori". The identification of useful semantic features is an open research issue [4]. Ways to obtain knowledge from perception and experience can use a dynamical system approach [14], behavioral models [26], or a developmental perspective [12].

The study of language together with action and perception has recently seen a large research effort focusing on integrating them within a unified cognitive model [6, 7, 21, 22].

C. *The PRAXICON database*

POETICON relies on the theoretical premise that meaning emerges from the integration of sensorimotor and symbolic representations, both of which contribute equally. POETICON developed a core database, called PRAXICON [19, 20], with the role of bridging a conceptual structure to low-level sensorimotor representations through the use of embodied concepts. The term *praxicon* [3], (from the Greek praxis = action), was introduced by Liepman in 1908 to define motor representations perceived and stored in the parietal cortex for motor production. The PRAXICONs are embodied concept representations of perceptual, motoric, linguistic or symbolic nature, perceived and stored in memory. The PRAXICON database is a 3rd normal form database centered on the definition of actions. It has been populated with 120,000 concepts from WordNet, and from results of cognitive experiments [22]. It represents concepts as entities having relations with other entities; a concept is retrieved by giving its representation in terms of language or of any other modality. [2] has demonstrated PRAXICON commanding the iCub robot.

Concepts in PRAXICON can have four types: Entity, Movement, Feature and Abstract. Abstract concepts are basic level only if they derive from entities or movement. For instance "cup" is an abstract concept, connected with various other concepts. It is either of type "container" or "dishware" and has a number of different realizations, like "coffee_cup", etc. It can be used as an artifact in the action "cup_with_dummy_artifact_the_dummy_object" (i.e. put something in a cup) and is also the container of something ("dummy_content_cup"). It has features size and shape.

Relations among concepts are in Table I and Fig. 1. The t*ype-token* relation defines a taxonomy among concepts; "A type-token B" means that A is a "father concept" for B. Relations can be arranged in chains, intersections (relations which have to hold at the same time) and unions (possible alternatives). A combination of relations can be *inherent* if it defines a necessary and sufficient condition for the entity it refers to (an intersection of three relations artifact-use, action-object, action-purpose is inherent for movement concepts). For example, the 'cut_dummyTool_bread'#movement concept is inherently related with a 'tool' entity concept. The reasoner searches for an 'entity' concept to fill in this variable for the specific concept. Many entities could take up such role (e.g. knife, hands etc.); therefore more solutions can be found.

TABLE I. THE PRAXICON RELATIONS

Between Entities	**Between Movements**	**Between Entity - Entity Descriptive Feature**
token-token	Token-token	has colour (has_hue, has_luminance, has_intensity)
type-token	type-token	has shape
part-whole	step-process	has size (has_length, has_height, has_width, has_depth)
artifact-user	symbol	has texture
people-place	**Between Entities-Movements**	has visual pattern
people_involved-institution	object-action	has condition (has_natural effect, has_anthropogenic effect)
container-content	means-action	has weight, has temperature
audience-performer	natural kind-purpose	**Between Movement - Movement Descriptive Feature**
prestate-entity	artifact-use	has essential actuator
product-producer	agent-action	has direction
product-retailer	place-event	has speed rate
symbol	trigger-event	has velocity
place-institution	action-location	has acceleration
material - entity	action-source	**Between Descriptive Features**
artifact-art	**Between Entity or Movement and Movement**	Type-token
animal-meat	result of action	Part-whole
	purpose of action	**Between Entity or Movement and Evaluative Feature**
	cause of action	Body_reaction-emotion
	cause of effect	Concept-aspect

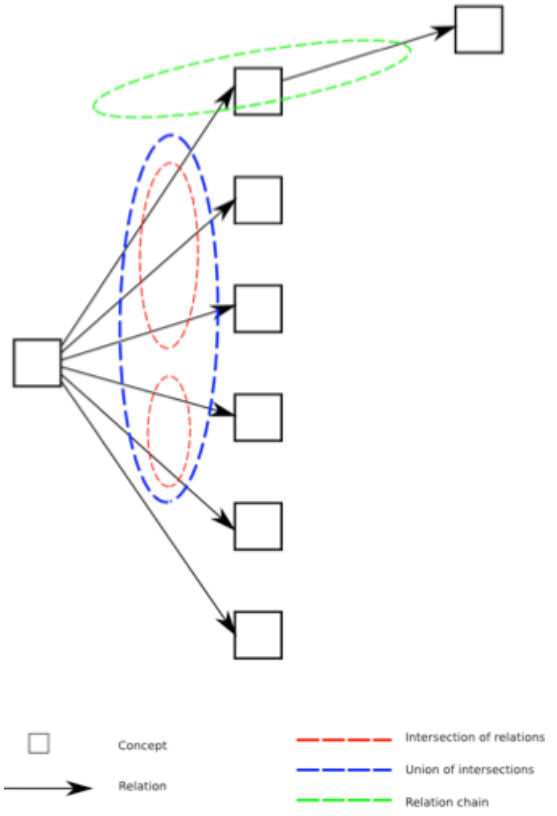

Fig. 1. Example of PRAXICON relations and their arrangement.

III. Logics representation of PRAXICON

The advantage in using a semantic approach is its ability to formalize and provide standard reasoning services to a conceptual system; anyway this approach has several conceptual and technical shortcomings and presents some new challenges. More specifically, the advantages of using semantic languages over data base search are:

- not only data instances but also data models can be queried, which means that the relationships among data can be discovered and extended by means of suitable queries;
- several knowledge bases can be queried with a single simple query, which can be seen as a graph instead than as a series of joins;
- it is possible to use structured and semi-structured data;
- it is not necessary to explicitly encode all the domain knowledge, since the reasoner can infer new facts.

Possible problems exist too in defining which aspects a concept is supposed to capture. Moreover, when knowledge is derived from "common sense" there might be a problem related to typicality and exceptions: DL do not allow for "partial inheritance", hence classes cannot be used directly for representing concepts.

Regarding the level of information to represent, a possible problem lies in the comparison between features: while a general solution involving concepts such as "comparison type" and "dimension of comparison" is more general, a DL-inspired solution (i.e. the use of specific properties) is more easily implemented. For example, when comparing two colors for deciding which one is "more red", in the first case the comparison would be of the type "more than" and the dimension would be the red channel in the RGB representation, while in the second case a moreRedThan property would be needed. The choice depends on the level at which the comparison is performed, in this case on whether the equality check is performed within the ontology (using SPARQL queries) or not.

We show now the construction of a knowledge base built on top of the PRAXICON database. The formalization of PRAXICON using DLs makes use of instances rather than classes for representing concepts because:

- Transforming thousands of concepts in classes would make it difficult to use a standard reasoner (the only reasoner which can handle such number is Snorocket, but it reasons on the OWL2 EL, where several constructs are not available).
- Partial queries on the concepts for checking the "degree of satisfaction" (such as in intersections of relations) should be possible.
- The path between two concepts and its length should be known (or decided beforehand), so it is possible using property paths on instances with SPARQL 1.1.
- It is easier to extend concept's definitions and to create/modify them automatically as relations with new instances.
- It is possible to "count" things (e.g. the number of relations which hold etc.)

So we added the following constraints to the original PRAXICON database:

- only *AbstractType* concepts can be basic or non basic;
- a concept is "fully abstract" if it is non basic and it does not have any origin;
- the type of a template derives from the type of the associated movement concept;
- only movements are templates;
- concepts having origin in an entity can have the status of a variable or a constant; concepts having origin in a movement can have the status of a constant or a template; concepts with no origin can only be constant.

The concepts in the DL implementation have been renamed using a hash of the representation present in the original database as obtained from WordNet; for example, the concept butter knife%1:06:00:: becomes butter knife 1477250444 in the ontology. Object properties are used to represent relations between concepts. The token-type relation is used as a subclass axiom: although it would need to be transitive, this is not necessary since SPARQL property paths are used. We added a role hierarchy with a general top property *relatedTo* as an ancestor property of all the PRAXICON relations, a *HasFeature* superproperty covering concept features (having entities as a domain and features as a range), and its inverse *FeatureOf*.

By adding domain and range axioms, subclasses of features are automatically created from the PRAXICON properties (e.g. the feature class Shape is created as the range of the *HasShape* property) so that PRAXICON feature concepts can be classified using more specific classes.

We used OWLIM 5.2[3] as semantic repository because it is suitable for big ABoxes, provides reasoning within OWL 2 RL and QL, and supports SPARQL 1.1 queries. The conversion from database to ontology has been done with the OWLAPI library[4]. The class structure has been fully implemented; only a part of the PRAXICON including stir, spread and cut actions, foods, kitchen tools and containers has been converted. For visualization and testing we used Protégé 4.2 (see Fig 2).

IV. The Experiments

To test the generalization capabilities, we ran several SPARQL queries against the ontology. Although we worked on the topic of semantic interpretation of sentences [25], and in POETICON a module is available for understanding a request expressed in natural language, for the time being we ignore the syntactic and semantic analysis of the sentence, thus focusing on the structure of the knowledge base. We suppose the available vision system could provide any level of detail to match entries in the knowledge base (e.g. it can find teaspoons, not only spoons). The experiments conducted are composed of:

- a verbal request for an action expressed as a triplet (*movement; tool; object*) where tool and object can be empty;

- a set of objects in the scene, whose type can be specific (e.g. butterknife, teaspoon) or generic (e.g. knife, spoon);

The decision is in the form of a triplet (*movement; tool; object*) defining the kind of movement to execute, the tool to use and the object affected. We do not currently score multiple solutions but we list all the possible solutions. Future implementations on a real robot would select the best solution according perhaps to external criteria, as kinematics constraints or reachability.

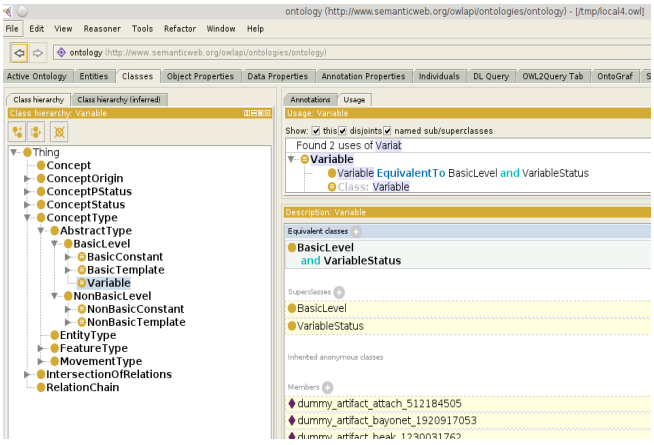

Fig. 2. The interface window of DL-PRAXICON; on the left the class hierarchy, on the right the variable characteristics.

[3] http://owlim.ontotext.com

[4] http://owlapi.sourceforge.net

In the following we illustrate the reasoning on five simple commands.

1. Spread the butter with a butter knife. The search is for the three lemmas spread, butter and butterknife; the instances found are: spread 456537824, butter 519714614, butter knife 1477250444. Looking for the possible intersections of relations containing them, the concept spread 971071545 related to the movement of "spreading the butter with a butter knife" is retrieved and the search ends with the resulting triplet (spread; butter; butterknife). No inference is needed.

2. Spread the butter with a knife. The search is the same as before, but no results are obtained. The search is extended using a generalization of the tool, thus finding out that such a movement exists when it is performed with a butterknife, making it a possible substitute. In this case the only information used is the taxonomy of the objects knife and butterknife (property paths with the relations TokenType and TypeToken). The result is then (spread; butter; knife).

3. Spread the butter with a dull/small object. Here dull is not an entity but a feature, thus the search has to find entity having such a feature. The feature concepts having as linguistic representations dull and small are searched, then concepts related to such features through any sub-property of the property HasFeature are searched. Finally the butterknife is found, along with the exact relations linking it to its features (HasShape and HasSize respectively). The result is then (spread; butter; butterknife).

4. Cut the turkey wing. This example is interesting because it shows how to generalize using a relation not entailing a feature but another entity; the turkey wing is a part of a turkey, so the relation HasPart is used. For finding a concept linking cut and turkey, turkey itself has to be generalized. The only concept found is *cut the staff of life with bread knife*, and the only common ancestor for turkey and staff of life is solid food, which is a non-basic level abstract concept, together with its descendants meat, bird and poultry. In this case the search in PRAXICON database would not go up all the chain, because it would stop to the first basic level abstract concept.

5. Stir the soup. The objects ontology we used for this last test was extended to contain 380 objects arranged in 19 categories, and able to reason on dimensions [24]. The tools available in the scene are only a teaspoon, a screwdriver, and a wrench, as in Fig. 3 and Table II that contains the measures for the relevant objects in the ontology. The movement concept containing both stir and soup is *Stir the soup with a spoon*. While the teaspoon is semantically close to the spoon, if the dimensions of the object are considered as more important, a screwdriver and a wrench could have a size "closer" to a serving spoon than a teaspoon. In this case, the comparison is checked on the metric features with a level of "closeness" that inversely depends on the difference in their features values. It is questionable how to assign the weights to semantic closeness and geometric closeness. Without this decision, as in this case, all the three available objects are reported as of possible use.

Fig. 3. The 3D models of the objects considered: teaspoon, screwdriver, wrench.

TABLE II. FEATURE VALUES OF THE OBJECTS

object	length	width	height
spoon	20.003	4.309	3.6225
	20.161	4.3432	3.6513
	21.273	4.5826	3.8525
teaspoon	14.922	3.2147	2.7025
	15.558	3.3515	2.8175
	15.875	3.4199	2.875
screwdriver	9.525	1.1009	1.0054
	22.542	2.6055	2.3794
	27.543	3.1835	2.9072
wrench	17.1	4.6077	1.3063
	21.9	5.9011	1.673
	23.1	6.2245	1.7646
	27.9	7.5178	2.1313

A further step of knowledge extraction from examples or the inclusion of common sense knowledge would be required to make the best choice. In this example it might help to know that for stirring movements, long objects are needed (information possibly extracted analyzing the semantic features of the objects used for stirring), so avoiding the use of a wrench, or that tools are dirty and dirty objects cannot be used with food (common sense knowledge, more difficult to obtain), so avoiding also using the screwdriver.

The graphical representation of the results of the queries is in Fig.4.

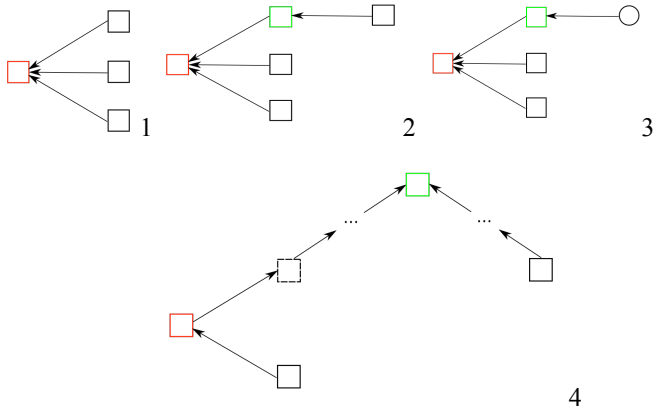

Fig. 4. Graphical results of the NL queries. In the first query the 3 concepts: butterknife, butter, and spread are directly found and related to the concept ‘spread the butter with a butterknife’ (in red). In the second query the knife, in green, is found as a generalization of butterknife. In the third query the knife is found from its properties HasShape and HasSize. In the fourth query the path is longer; solid food (in green) is related to turkey wing on one side through generalization (turkey wing, turkey, poultry, solid food), and on the other side it is also connected to staff of life through (staff of life , bread, solid food). Staff of life is the concept used in the cut movement concept (in red).

V. CONCLUSIONS

In this paper we have illustrated some problems encountered when trying to use generic natural language sentences to ask for robot actions. Our solution explores the feasibility of DLs to provide an internal representation and a reasoning tool. To deal with the embodiment problem we observed that the extraction of knowledge from sensing starts offering solutions, albeit not yet practical for any domain. The action planning problem is not considered here, as literature offers solutions, as for instance [2].

Here we wanted to estimate the feasibility of a DL representation, considering that the DL technologies still have representation and scalability problems. We started from an available large robot-oriented database, the PRAXICON, and transformed it into ontology. Doing that the following considerations emerged:

- The use of classes for representing concepts is not suitable for several reasons: the typicality/exceptions issues, and the difficulty in building, storing and reasoning on expressive axioms when they are too many.
- Instance checking, and instance retrieval are the main reasoning needed.
- Data-types properties are used for linguistic representations.
- Even though the technology to store large ontologies is still lacking, OWLIM 5.2 was capable of performing inference on OWL constructs while supporting a very high number of instances. Scalability might be a concern only up to a certain extent; a robot typically does not have an unlimited number of skills, therefore it does not need to store and use huge amounts of information.
- The queries on the ontology using SPARQL 1.1 take advantage of property paths for taxonomic relations.

The integration of other sources of information carrying different levels of representation is under investigation: we provided an example using an external ontology about objects, and we are currently studying how to integrate a grasp ontology too. In the future, to extend this case study to a real application, several improvements should be made, in particular on the representation side, for formalizing “relations between relations", and to include common knowledge statements.

Putting our results in a perspective, we can conclude that DLs representation can be an extra force in simulators too. Today the development of robot applications is mainly done with the help of simulators. Considering the still open problems in using simulators instead of reality, our approach may improve the simulation process in dividing what depends on the world model and the cognitive capabilities of the robot from what depends on the robot movements and perceptions:

- using ontologies, the simulation is done in a hierarchical way, from the models to the behaviors;
- initially the model of the scene and the actions ontologies are checked for understanding whether something is missing or wrongly described. For instance, something may be wrongly described in the knowledge model if the robot

decides to use a wrench to stir the soup, while a servingspoon is available;

- after understanding the command, the obtained triplet indicates the behavior to use; as in [2] the triplet can subgoal for the planner, or can call an implemented skill;
- the second level simulation is where the behavior is executed and where real time sensing occurs, as usual.

Although the semantic approach in robotics deserves much interest, a specific use of it to allow programming robots in natural language has not yet been fully explored.

ACKNOWLEDGMENT

The use of the PRAXICON has been possible thanks to a kind invitation from K. Pastra to the first author of this paper to temporary join the POETICON++ project.

REFERENCES

[1] J. R. Anderson, D. Bothell, M. D. Byrne, S. Douglass, C. Lebiere, and Y. Qin, "An integrated theory of the mind". Psychological Review, 111:1036-1060, 2004.

[2] A. Antunes, L. Jamone, G. Saponaro, A. Bernardino, R. Ventura, "From Human Instructions to Robot Actions: Formulation of Goals, Affordances and Probabilistic Planning", Proc IEEE ICRA 2016.

[3] M. A. Arbib, How the Brain Got Language: The Mirror System Hypothesis, Oxford University Press, USA, Apr 2012.

[4] F. Baader, D. Calvanese, D.L. McGuinness, D. Nardi, and P.F. Patel-Schneider eds., The description logic handbook: theory, implementation, and applications, Cambridge University Press, New York, NY, USA, 2003.

[5] L. W. Barsalou, "Perceptual symbol systems", The Behavioral and brain sciences, 22(4):577-609, 1999.

[6] D. Caligiore and M. Fischer, "Vision, action and language unified through embodiment", Psychological Research, January 2013, Volume 77, N 1, p 1–6.

[7] A. Cangelosi and M. Schlesinger. Developmental Robotics: from babies to robots, MA: MIT Press, 2015.

[8] P. F. Dominey, A. Mallet, E. Yoshida,. " Progress In Programming the HRP-2 Humanoid Using Spoken Language", IEEE ICRA, Roma, Italy, 10-14 April 2007, p 2169-2174, 2007.

[9] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth, "Describing objects by their attributes", 2009 IEEE Conference on Computer Vision and Pattern Recognition, p 1778-1785, 2009.

[10] A. M. Franchi, L. Sernicola, G. Gini, "Linguistic Primitives: a New Model for Language Development in Robotics", E. Tuci et al. (Eds.): SAB 2016, Springer International Publ. Switzerland, LNAI 9825, p 207–218, 2016.

[11] G Gini, M Gini, M Somalvico. "Emergency recovery in intelligent robots" , Proc. 5th ISIR, Chicago, Illinois, p 339-358, 1975.

[12] S. Griffith, J. Sinapov, V. Sukhoy, and A. Stoytchev, "A behaviorgrounded approach to forming object categories: Separating containers from noncontainers", IEEE T. Autonomous Mental Development, 4(1):54-69, 2012.

[13] S. Harnad., "The symbol grounding problem". In CNLS'89, pages 335:346, North-Holland Publishing Co, Amsterdam, The Netherlands 1990.

[14] B. Kuipers, "Drinking from the _rehose of experience", Artif. Intell. Med., 44(2):155-170, Oct. 2008.

[15] J. E. Laird, A. Newell, and P. S. Rosenbloom, "SOAR: an architecture for general intelligence", Artif. Intell., 33(1):1-64, Sept. 1987.

[16] D. Lenat, "CYC: A large-scale investment in knowledge infrastructure". Comm of the ACM, 38(11):33-38, November 1995.

[17] H. Liu and P. Singh. "Conceptnet: A practical commonsense reasoning toolkit", BT Technology Journal, 22(4):211-226, 2004.

[18] K.F. MacDorman, "Grounding symbols through sensorimotor integration", J.Robot. Soc. Jpn. 17, 20–24 ,1999.

[19] K. Pastra, P. Dimitrakis, E. Balta, and G. Karakatsiotis. "PRAXICON and its language-related modules", In SETN, 2010.

[20] K. Pastra and Y. Aloimonos, "The minimalist grammar of action", Philosophical Transactions of the Royal Society B, volume 367, p 103-117, January 2012.

[21] K. Pastra " Autonomous acquisition of sensorymotor experiences: any role for language?", IEEE Computational Intelligence Society, Newsletter on Autonomous Mental Development, vol. 10, n 2, 2013.

[22] A. Vatakis, K. Pastra, and P. Dimitrakis, "Acquiring object affordances through touch, vision, and language", In 13th International Multisensory Research Forum. June 2012.

[23] N. Vitucci, "Autonomous object manipulation: A semantic-driven approach", Proceedings IJCAI, p 2858-2859, 2011.

[24] N. Vitucci, "Description logics and semantic query languages in robotic applications", Phd Thesis, Politecnico di Milano, 2013.

[25] N. Vitucci, M. A. Neri, R. Tedesco, and G. Gini, "Semanticizing Syntactic Patterns in NLP Processing Using SPARQL-DL Queries", Proc OWLED, pag 1-8, http://ceur-ws.org/Vol-849/, 2012.

[26] E. Yee, S. Huffstetler, and S. L. Thompson-Schill, "Function follows form: activation of shape and function features during object identification". Journal of experimental psychology. 140(3): 348-363, 2011.